8-2003

Superquadric representation of scenes from multi-view range data

Yan Zhang

To the Graduate Council:

I am submitting herewith a dissertation written by Yan Zhang entitled "Superquadric representation of scenes from multi-view range data." I have examined the final electronic copy of this dissertation for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy, with a major in Electrical Engineering.

Mongi A. Abidi, Major Professor

We have read this dissertation and recommend its acceptance:

Accepted for the Council:

Carolyn R. Hodges

Vice Provost and Dean of the Graduate School

(Original signatures are on file with official student records.)

To the Graduate Council:

I am submitting herewith a dissertation written by Yan Zhang entitled "Superquadric Representation of Scenes from Multi-View Range Data". I have examined the final paper copy of this dissertation for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy, with a major in Electrical Engineering.

M. A. Abidi

Mongi A. Abidi, Major Professor

We have read this dissertation
and recommend its acceptance:

David B. Koch

Acceptance for the Council:

Vice Provost and Dean of Graduate Studies

SUPERQUADRIC REPRESENTATION OF SCENES FROM MULTI-VIEW RANGE DATA

A Dissertation
Presented for the
Doctor of Philosophy
Degree
The University of Tennessee, Knoxville

Yan Zhang
August 2003

Acknowledgments

I would like to thank the people who have helped and supported me in completing this work. First of all I would like to thank my supervisor Dr. Mongi Abidi for his support, patience, and guidance during these years of my study at UTK. Also, I would like to thank the other members of my dissertation committee, Dr. Collins, Dr. Koch, Dr. Qi, and Dr. Roberts, for their interests in this work and their insightful advice to this dissertation.

I am very grateful to Dr. Koschan and Dr. Paik for their invaluable suggestions to my research and this dissertation. Special thanks to Dr. David Page for our inspiring conversations and his many helpful comments on this dissertation.

I would also like to thank the faculty, staff and students in the IRIS laboratory who created an excellent environment where I have enjoyed working. I am indebted to Vicki Courtney-Smith for her helping with my various administrative needs, to Mark Mitckes for his proofreading of my outgoing documents, and to the fellow students including Brad Grinstead, Umayal Chidambaram, Tak Motoyama, Justin Acuff and many others for their kind help.

Last but not least, I want to express gratitude to my family. Thanks to my parents Mingyu Zhang and Shuiyue Xu for their consistent support during these academic years. Huge thanks to my husband, Zhong Tang, for his love and encouragements and for his understanding and sharing everything in my life.

Abstract

Object representation denotes representing three-dimensional (3D) real-world objects with known graphic or mathematic primitives recognizable to computers. This research has numerous applications for object-related tasks in areas including computer vision, computer graphics, reverse engineering, etc. Superquadrics, as volumetric and parametric models, have been selected to be the representation primitives throughout this research. Superquadrics are able to represent a large family of solid shapes by a single equation with only a few parameters.

This dissertation addresses superquadric representation of multi-part objects and multi-object scenes. Two issues motivate this research. First, superquadric representation of multi-part objects or multi-object scenes has been an unsolved problem due to the complex geometry of objects. Second, superquadrics recovered from single-view range data tend to have low confidence and accuracy due to partially scanned object surfaces caused by inherent occlusions. To address these two problems, this dissertation proposes a multi-view superquadric representation algorithm. By incorporating both part decomposition and multi-view range data, the proposed algorithm is able to not only represent multi-part objects or multi-object scenes, but also achieve high confidence and accuracy of recovered superquadrics. The multi-view superquadric representation algorithm consists of (i) initial superquadric model recovery from single-view range data, (ii) pairwise view registration based on recovered superquadric models, (iii) view integration, (iv) part decomposition, and (v) final superquadric fitting for each decomposed part.

Within the multi-view superquadric representation framework, this dissertation proposes a 3D part decomposition algorithm to automatically decompose multi-part objects or multi-object scenes into their constituent single parts consistent with human visual perception. Superquadrics can then be recovered for each decomposed single-part object. The proposed part decomposition algorithm is based on curvature analysis, and includes (i) Gaussian curvature estimation, (ii) boundary labeling, (iii) part growing and labeling, and (iv) post-processing. In addition, this dissertation proposes an extended view registration algorithm based on superquadrics. The proposed view registration algorithm is able to handle deformable superquadrics as well as 3D unstructured data sets. For superquadric fitting, two objective functions primarily used in the literature have been comprehensively investigated with respect to noise, viewpoints, sample resolutions, etc. The objective function proved to have better performance has been used throughout this dissertation.

In summary, the three algorithms (contributions) proposed in this dissertation are generic and flexible in the sense of handling triangle meshes, which are standard surface primitives in computer vision and graphics. For each proposed algorithm, the dissertation presents both theory and experimental results. The results demonstrate the efficiency of the algorithms using both synthetic and real range data of a large variety of objects and scenes. In addition, the experimental results include comparisons with previous methods from the literature. Finally, the dissertation concludes with a summary of the contributions to the state of the art in superquadric representation, and presents possible future extensions to this research.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

This dissertation addresses the problem of representing real-world objects with high-level, compact primitives. In 3D computer vision, objects are usually scanned by laser range sensors or stereo cameras, and 3D coordinates of points on object surfaces can be obtained. A large number of points (dense sample) are often desired in many computer vision tasks such as surface reconstruction. By representing a huge number of unstructured points with compact primitives, storage, visualization and manipulation costs of the objects can be significantly reduced. Such an object representation step is indispensable to robot navigation systems, especially to an object recognition task. In addition, representing objects with mathematic or graphic primitives that can be recognized by computers is crucial to many applications in reverse engineering, computer graphics, etc. This dissertation focuses on superquadric representation of multi-part objects and multi-object scenes. We define a multi-part object as an object consisting of multiple parts according to part theories in human perception. One of the part theories is the transversality regularity (Guillemin and Pollack, 1974), for which part boundaries are distinguished by negative curvatures. For instance, a teapot consists of five parts while a computer mouse contains only one part according to the transversality regularity. A multi-object scene denotes a scene consisting of multiple separate or articulated objects. These definitions are used in this dissertation.

In the following sections, we first describe the problem statement and motivations for this research in Section 1.1. Section 1.2 briefly introduces object representation and primarily used primitives. We summarize the contributions of this dissertation in Section 1.3. Finally, the organization of this dissertation is provided in Section 1.4.

1.1 Problem Statement and Motivations

A major application of this research is vision-based robotics for cleaning up waste in a hazardous industrial environment, for example, a radiological waste area in a typical DOE task (National Energy Technology Laboratory, deactivition & decommissioning focus area, 2002) as shown in Fig. 1.1. In such a hazardous environment, robots are needed to navigate, collect useful data, and to recognize and manipulate objects as a human would. Computer vision

Figure 1.1: A radiological waste area in a typical DOE task (National Energy Technology Laboratory, deactivition & decommissioning focus area, 2002).

tasks involved in such robotic vision systems include scene building, scene description, 3D data visualization, etc. as shown in Fig. 1.2.

The scene description step describes real scenes using primitives that are acceptable to computers in a compact manner. It directly facilitates succeeding tasks including object recognition, data visualization, data communication, etc. Our research, object representation, falls into the scene description category. Another important application of this research is reverse engineering (RE), i.e., creating a computer-aided design (CAD) model of a real-world object. Traditional methods of RE with coordinate measuring machines (CMM) are often tedious and time consuming. In computer vision and image processing, recent advances in 3D free form scanning, however, have led to efficient, accurate, and fast laser-based systems that rapidly generate high fidelity computer models of existing automotive parts. The automation of RE, or computer-aided reverse engineering (CARE), impacts the design process in a similar fashion, especially with a collaborative, distributed design team. CARE allows electronic dissemination of as-built parts for comparison of original designs with manufactured results. Additionally, CARE allows construction of CAD models of existing parts when such models no longer exist, as when parts are out of production (Thompson et al., 1999). Of particular interest is the Mobile Parts Hospital initiative within the U. S. Army Tank-Automotive and Armament Command (TACOM). The vision for the parts hospital is an emergency manufacturing unit that is designed for frontline deployment. The three main areas of CARE are data acquisition, model reconstruction, and post processing. The object representation falls into both model reconstruction and post processing. For instance, for an automotive part shown in Fig. 1.3, object representation aims to represent unstructured data points of the object with appropriate primitives which can correctly capture the geometry of the object.

In summary, we seek **object representation methods and primitives** that are able to represent real-world objects and scenes efficiently. Considering industrial applications of this research, we are only concerned with representing the geometry of rigid objects. Based on our literature review, the superquadric (SQ) is selected as our representation primitive due to its

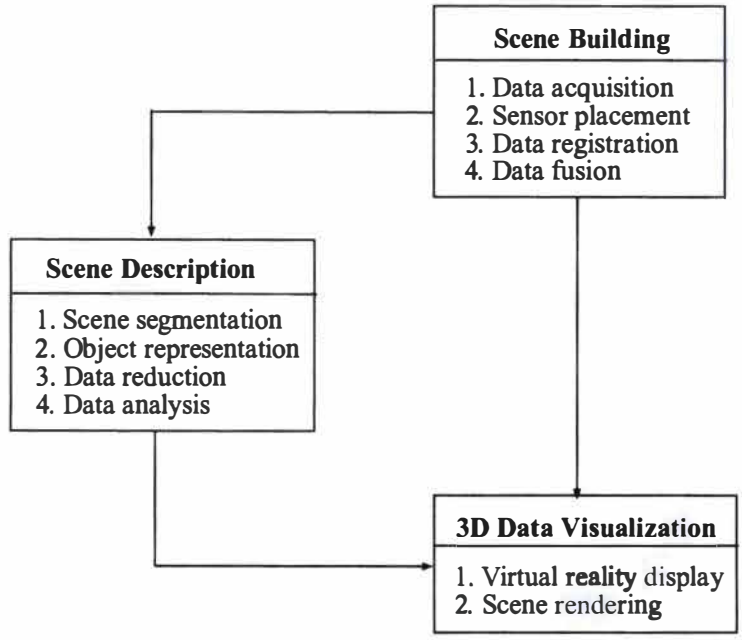


Figure 1.2: Research topics involved in a robotic vision task (Imaging, Robotics and Intelligent Systems Laboratory, 2002).

Figure 1.3: A distributor cap.

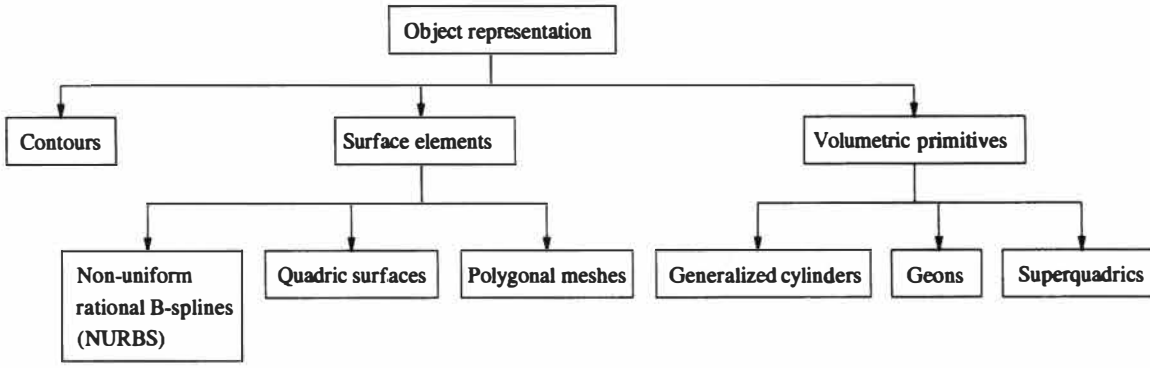


Figure 1.4: Various levels of primitives used in object representation.

compact and powerful representation abilities. As to superquadric representation strategies, we propose a multi-view superquadric representation algorithm to address two problems in the literature. First, Superquadric representation of multi-part objects or multi-object scenes has been an unsolved problem due to the complex geometry of objects. Second, superquadrics recovered from single-view range data tend to have low confidence and accuracy due to partially scanned object surfaces caused by inherent occlusions. By incorporating both part decomposition and multi-view range data, the proposed multi-view superquadric representation algorithm is able to not only represent multi-part objects or multi-object scenes, but also achieve high confidence and accuracy of recovered superquadrics.

1.2 Overview of Object Representation

One of the ultimate goals of computer vision is to achieve intelligent interactions between computers or other automated devices and the real world using various types of images. Images consisting of thousands of individual elements (pixels) need to be represented in a compact manner for many tasks in computer graphics, reverse engineering, image communication, and computer vision. Object representation tackles this problem, and describes real-world objects with compact graphic or mathematic models that can be manipulated by computers. This representation needs to reflect the intrinsic structure of objects, (i.e., distinct objects should have distinct representations), and be compact, accurate and efficient (Jaklič et al., 2000) as well. Object representation is an indispensable step in object- or model-oriented tasks including CAD modeling, object recognition, autonomous navigation, etc.

The shape of an object can be represented by three levels of primitives: volumetric primitives, surface elements, and contours according to the complexity of the primitives (Bajcsy et al., 1990). A taxonomy of primitives used in object representation is depicted in Fig. 1.4. The primitive selected to describe an object depends on the complexity of the object and the tasks involved. A comparison between various levels of primitives and their characteristics (Dorai and Jain, 1997) is illustrated in Fig. 1.5. Among the various levels of primitives, contours have low granularity, and are too local to capture or take advantage of the overall structure

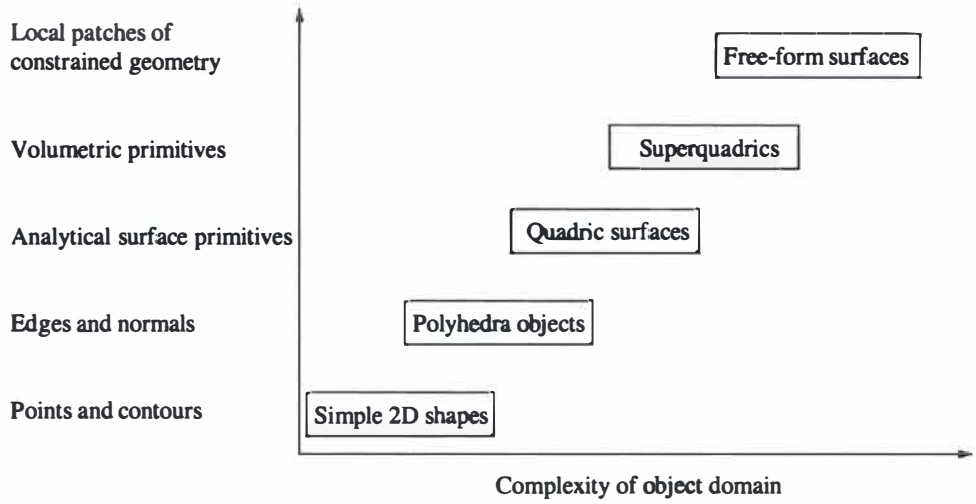


Figure 1.5: Representation primitives vs. object shape complexity (Dorai and Jain, 1997).

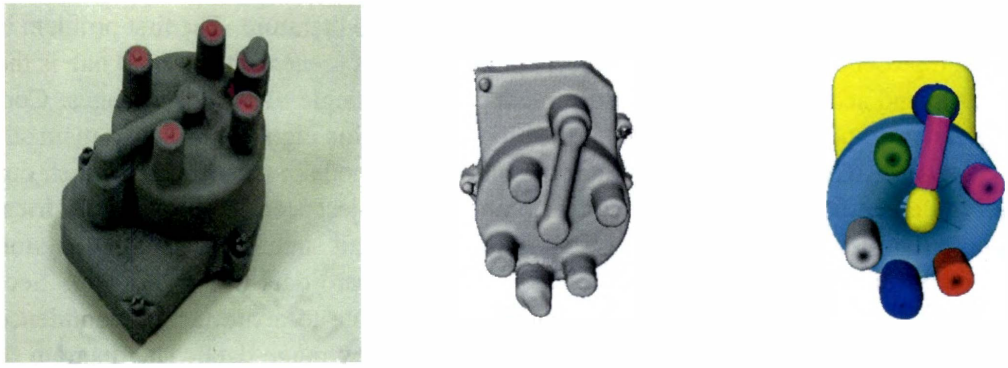
of complex objects. However, they have the remarkable capability of describing local details, and are indispensable for providing local information in a high-level representation task. On the other hand, surface elements and volumetric primitives, briefly described as follows, have attracted more attention due to their abilities to represent a wide range of objects efficiently.

Surface Elements. A solid object can be represented by its bounding surfaces. This description can vary from simple triangular patches to visually appealing structures such as non-uniform rational B-splines (NURBS), which are popular in geometric modeling. An example of a teapot represented by 3D triangle meshes is illustrated in Fig. 1.6. A considerable amount of effort in computer vision and computer graphics has been devoted to describing complex surfaces as piecewise continuous patches from 3D data, which is typically defined as a 3D surface reconstruction problem (Campbell and Flynn, 2001). Surface reconstruction aims to create smooth surface descriptions from dense or sparse depth measurements that are typically corrupted by outliers and noise. Surface reconstruction of objects with arbitrary topology has usually been performed on triangular meshes because polygonal meshes provide a simple piecewise planar surface representation that gives a first order approximation without constraints on topology (Hilton et al., 1996). In addition, a triangulated mesh can be easily constructed for a single range image. Existing integration algorithms aim to merge constrained triangulations of individual range images. As middle-level primitives, meshes have the advantages of representing arbitrary shapes with various levels of details by adjusting mesh resolutions. The disadvantages include high computation and storage costs caused by the huge number of representation elements. For instance, the mesh for the teapot shown in Fig. 1.6 consists of approximately 3,000 vertices and 6,000 triangles. Therefore, these surface representations, when expressed by a set of triangle or polygon meshes, can only be sufficient for applications that need simple

Figure 1.6: A teapot model represented by 3D triangle meshes. The mesh consists of 3,042 vertices and 6,026 triangles. This model is a reconstruction from the Hughes Hoppe at Microsoft Research.

geometric representation such as a visualization task. High-level tasks such as object recognition needs more structural primitives to provide compacter information. In addition, tasks that require fast or even real-time model manipulations need more efficient primitives.

Volumetric Primitives. As the highest-level representation, volumetric primitives are the most compact, and represent the most intuitive decomposition of an object into parts. Unlike surface elements, volumetric primitives represent solids or volumes instead of surfaces. Models composed of volumetric primitives can easily support part articulation and, at the structural level, are insensitive to dimensional changes in the parts (Bajcsy et al., 1990). Therefore, volumetric primitives are also called part models. These part-level characteristics enable volumetric primitives to support object manipulation, functional-based object recognition, and other high-level activities. Several volumetric primitives for modeling object parts have been proposed in the literature. The most used include generalized cylinders, geons, superquadrics, and many other primitives (Shirai, 1987). These part-level models can be classified into two categories: qualitative and quantitative (parametric) models. Generalized cylinders are the first dedicated part-level models in computer vision (Binford, 1971). Generalized cylinders have influenced much of the model-based vision research in the past two decades. A generalized cylinder is formed from a volume by sweeping a two-dimensional set along an arbitrary space curve. Geons consist of a set of solid blocks which are derived from generalized cylinders. They are proposed by Biederman (Biederman, 1985) in the context of human perception. The set of geons consists of 36 primitives obtained by changing axis shape, cross-section shape, cross-section sweeping function, and cross-section symmetry of generalized cylinders. Superquadrics appeared in computer vision as an answer to some of the problems with generalized cylinders (Pentland, 1986). As a subclass of generalized cylinders, superquadrics are a family of geometric solids



(a) (b) (c)

Figure 1.7: Superquadric representation for a distributor cap. This mesh is a reconstruction from multiple range images using the IVP Ranger System. The mesh consists of 58,975 vertices and 117,036 triangles while the decomposition consists of 13 parts. The original mesh is segmented using our part decomposition algorithm. Each of the decomposed parts is fitted to a superquadric model. Each superquadric is rendered in 3D using quad meshes based on recovered superquadric parameters. (a) Photograph of original object, (b) rendering of original mesh, and (c) rendering of superquadrics for decomposed parts.

which can be interpreted as a generalization of basic quadric surfaces and solids. With only a few parameters, superquadrics can represent a large variety of standard geometric solids as well as smooth shapes, which makes superquadrics much more convenient for object representation. An example of a multi-part object represented by superquadrics is illustrated in Fig. 1.7. A survey of volumetric primitives and corresponding representation methods is presented in Chapter 2.

1.3 Contributions

This dissertation presents three major contributions in superquadric representation, part decomposition, and view registration. The proposed algorithms extend the state of the art in these three fields. In particular, we have proposed a **multi-view superquadric representation algorithm** of which the advantage is representing multi-part objects or multi-object scenes with superquadrics in high fidelity and accuracy. Within the multi-view superquadric representation framework, we have presented a **3D part decomposition algorithm** based on transversality regularity to automatically decompose multi-part objects into their constituent single parts. Finally, we have extended a **view registration algorithm** based on superquadrics to register multi-view 3D range data in a pairwise manner. The three contributions proposed in this dissertation are summarized as follows.

Multi-view superquadric representation. The most significant contribution is the development of a multi-view superquadric representation algorithm. The multi-view superquadric

representation algorithm addresses two unsolved problems in the literature. The first problem is superquadric representation of multi-part objects or multi-object scenes. The second one is the low confidence and accuracy of superquadrics recovered from single-view range images. Correspondingly, our proposed multi-view superquadric representation algorithm has two main advantages compared to existing superquadric representation methods in the literature. First, our algorithm is able to efficiently represent multi-part and multi-object scenes with superquadrics. The incorporation of a part decomposition step turns the difficult superquadric representation of multi-part objects or multi-object scenes into a straightforward data fitting problem. Second, our proposed superquadric representation algorithm is able to achieve high confidence and accuracy of recovered superquadrics by utilizing multi-view range data. The diagram of the proposed multi-view superquadric representation algorithm is illustrated in Fig. 1.8. This algorithm is presented in Chapter 3.

3D part decomposition. The second major contribution is the development of a 3D part decomposition algorithm within the proposed multi-view superquadric representation framework as shown in Fig. 1.8. The proposed part decomposition algorithm is based on the principal of transversality regularity (Hoffman and Richards, 1984). Different from surface or region segmentation, the proposed part decomposition algorithm segments multi-part objects into their constitute parts that are recognizable to human perception systems. This algorithm takes surfaces represented by triangular meshes as input. Compared to the part decomposition algorithm (Wu and Levine, 1997) using simulated electrical charge distributions, the proposed algorithm can be easily implemented, and has reliable performances. The most significant advantage of the proposed part decomposition algorithm is that it directly facilitates the superquadric representation of multi-part objects or multi-object scenes. The proposed part decomposition algorithm is presented in Chapter 4.

View registration based on superquadrics. The third contribution is a 3D view registration algorithm based on superquadrics. This algorithm extends the range image registration algorithm (Jaklič et al., 2000) in three aspects. First, we have derived the first- and second-order inertial moments for tapered superquadrics, which enable tapered superquadrics to be registered with the extended algorithm. Second, the original 2D range image registration algorithm is extended in our algorithm to register 3D surfaces. Finally, the extended and improved view registration algorithm can register scenes consisting of complex backgrounds and objects under the assumption that there is at least one object in the scene which can be represented by a superquadric, and the object is visible (scannable) from all views to be registered. Compared with correspondence-based view registration methods, our algorithm needs much less overlap between views, and is less time-consuming. As view registration is indispensable to computer vision tasks involving multi-view data, the proposed view registration algorithm is crucial to our multi-view superquadric representation algorithm. The proposed view registration algorithm is presented in Chapter 5.

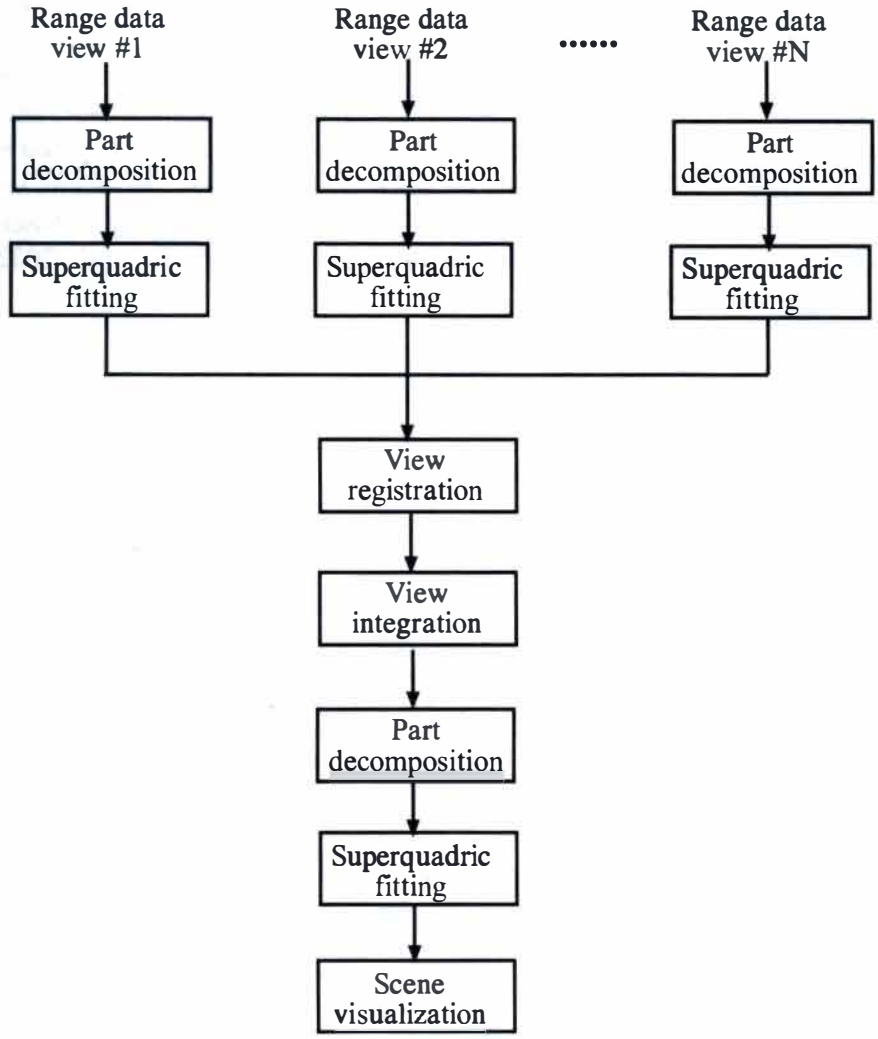


Figure 1.8: Diagram of multi-view superquadric representation algorithm. Three-dimensional triangle meshes for the range images are used as input. Part decomposition is utilized for multi-part objects or multi-object scene. The number of views depends on the complexity of the objects or scenes and the tasks involved. View integration is performed by RapidForm software.

1.4 Organization of the Dissertation

The remainder of this dissertation is organized as follows. Chapter 2 presents a literature review for each contribution. In addition, this chapter reviews the theoretical background on regular (undeformed) superquadrics, global deformations, free-form deformations performed on superquadrics, and several object functions in superquadric recovery. We propose our multi-view superquadric representation algorithm in Chapter 3. Chapter 4 presents our 3D part decomposition algorithm. The view registration algorithm based on recovered superquadrics is proposed in Chapter 5. Chapters 6 and 7 present experimental results for supplementary issues and proposed contributions. The experimental results demonstrate the efficiency of our proposed algorithms on a wide variety of objects and scenes. We conclude this dissertation and present several possible future research topics in Chapter 8. Most of the material presented in this dissertation has been published in (Boughorbel et al., 2003; Zhang et al., 2002a; Zhang et al., 2002b; Zhang et al., 2001; Zhang et al., 2003).

Chapter 2

Literature Review and Background on Superquadrics

This chapter presents a review of the research literature and background on superquadrics. Theoretical background on superquadrics is crucial to superquadric recovery. During a superquadric fitting process, the objective function used to measure how well the superquadric model fits the input data involves both definitions and geometric characteristics of superquadrics. Section 2.1 presents a brief survey of surface primitives and surface reconstruction methods applying polygonal meshes. Object representation algorithms using major volumetric primitives are also reviewed to establish our choice of superquadrics. In Section 2.2, we discuss superquadric representation methods in terms of input data types, recovery strategies, and complexities of objects involved. In order to represent multi-part objects and obtain high confidence, this dissertation proposes a multi-view superquadric representation algorithm. Within this algorithm, part decomposition and view registration are two key steps. Therefore, Sections 2.3 and 2.4 review part decomposition and view registration algorithms respectively. In Section 2.5, definitions and characteristics of regular superquadrics are introduced. Superquadrics with global deformations and free-form deformations (FFDs) are introduced. In addition, this section describes four objective functions in the literature for superquadric recovery and explores their performances. Finally, Section 2.6 concludes this chapter with a summary of the state of the art in superquadric representation.

2.1 Primitives and Object Representation

2.1.1 NURBS and Quadric Surfaces

The three major surface primitives include non-uniform rational B-splines (NURBS), quadric surfaces, and polygonal meshes according to the taxonomy of primitives illustrated in Fig. 1.4. As one of the most common parametric surface formulations, NURBS are defined as

$$S(u,v) = \sum_{i=0} \sum_{j=0} B^h_{i,j} N_{i,k} M_{i,l}(v), \tag{2.1}$$

where $N_{i,k}(u)$ and $M_{i,l}(v)$ are the B-spline basis functions of order k and l. $B^h_{i,j}$ represents the homogeneous coordinates of the control points (Koivunen and Bajcsy, 1995). NURBS are tensor-product forms. It has been shown that natural quadrics such as spheres, cylinders, and cones admit exact representation as NURBS. The homogeneous coordinates make this representation flexible.

A quadric surface is defined as a graph of an equation of second degree in three-dimensional Cartesian coordinates (Thomas and Finney, 1979). Common quadric surfaces include sphere, cylinder, elliptic cone, ellipsoid, etc. A quadric surface is formulated as

$$Ax^2 + By^2 + Cz^2 + 2Dxy + 2Eyz + 2Fxz + 2Gx + 2Hy + 2Jz + K = 0, \qquad (2.2)$$

or $P^TQP = 0$, where

$$P = \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}, Q = \begin{bmatrix} A & D & F & G \\ D & B & E & H \\ F & E & C & J \\ G & H & J & K \end{bmatrix}.$$

NURBS and quadric surfaces have not been widely used in computer vision due to their parametric formulations. In particular, it is difficult to make a surface defined on a parametric rectangle fit an arbitrary region on the surface of an object. This problem necessitates the use of trimming curves, which are not always unique and not generally detectable in imagery (Besl, 1990; Campbell and Flynn, 2001). Moreover, the homogeneous control points are not generally detectable nor unique. The completeness of parametric forms makes them useful as a source of an initial object specification, from which a polygonal mesh or other representations can be generated and employed in a computer vision system (Campbell and Flynn, 2001).

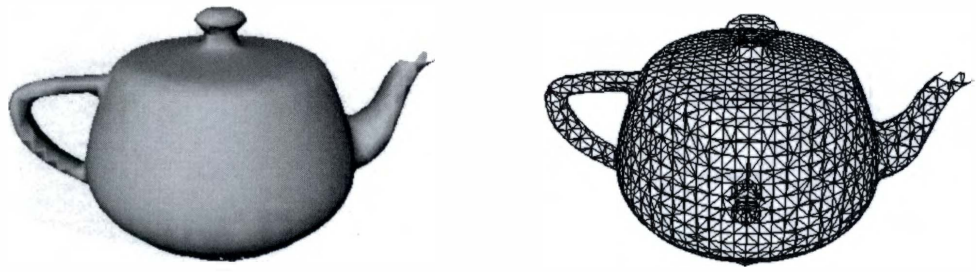
2.1.2 Polygonal Meshes and Surface Reconstruction

A polygonal mesh is a popular surface representation for 3D objects. A shared vertex-list notation is common for such surface representations. An object represented by polygonal meshes is defined by a pair of ordered lists (Campbell and Flynn, 2001) as

$$\mathcal{O} = \langle \mathcal{P}, \mathcal{V} \rangle, \qquad (2.3)$$

where $\mathcal{V} = v_1, ..., v_{N_v}$ is a list of N_v three-dimensional vertices $v_i = (x_i, y_i, z_i)^T$, and $\mathcal{P} = p_1, ..., p_{N_p}$ is a list of polygons, each specified as a list of vertex indices: $p_i = v_{i,1}, ..., v_{i,nv_i}$. The mesh consists strictly of triangles when $nv_i = 3$ for all i. The guaranteed convexity of triangles allows simple rendering algorithms to be used by synthetic images of models. A variety of techniques, commonly called polygonization methods, exist for generating polygonal mesh approximations from other geometric primitives such as implicit surfaces (Ning and Bloomenthal, 1993), parametric surfaces (Krishnamurthy and Levoy, 1996), and isosurfaces (Lorenson and Cline, 1987).

Polygonal meshes have a long history in computer graphics, and have become increasingly popular as surface representation primitives in computer vision over the last decade. This



(a) (b)

Figure 2.1: A teapot model represented by triangle meshes. This mesh is a reconstruction from Hughes Hoppe at Microsoft Research. The mesh consists of 3,042 vertices and 6,026 triangles. (a) Rendering of original mesh and (b) triangle meshes without shading.

increase in popularity is due to several factors including advances in computer hardware techniques and an increase in the popularity of dense range sensors, which produce rectangular arrays of 3D points that can easily be triangulated into meshes. Meshes can approximate surfaces of arbitrary objects with high accuracy given sufficient space to store the representation. Fig. 2.1 shows a teapot represented by triangle meshes. Triangle meshes are the most commonly used polygonal meshes. Triangulation of irregular data points is an example of data interpolation. The best-known technique is called the Delaunay Triangulation, which can be defined in two, three, or more space dimensions. A well-known algorithm to compute the Delaunay Triangulation can be found in (Preparata and Shamos, 1985). One problem with Delaunay Triangulation is that it triangulates the convex hull of the point sets. Constrained Delaunay Triangulation (Faugeras, 1993) can be a solution to this problem. The simplex mesh is another surface primitive that has a dual underlying graph with triangulations and has constant connectivity between vertices. Unlike triangulations, discrete geometric entities, such as curvature or normal vectors, can be applied easily to control the shape of simplex meshes. Delingette (Delingette, 1997) used simplex meshes to describe deformable objects. In this approach a model is first initialized manually, and the topology of the model is next modified by creating holes or increasing its genus. Finally, an iterative adaption algorithm decreases the distance between the model and the data. With this algorithm, the deformable simplex meshes were applied to represent objects from both medical images and range data (Delingette, 1997).

A considerable amount of effort in computer vision and computer graphics has been devoted to describing complex surfaces as piecewise continuous patches from 3D data, which is typically defined as a 3D surface reconstruction problem (Campbell and Flynn, 2001). Surface

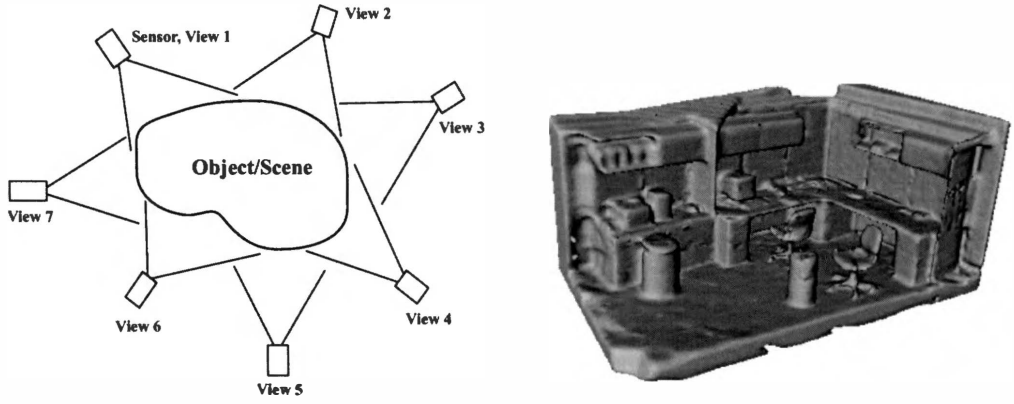


Figure 2.2: An office scene represented by triangle meshes. The mesh is reconstructed from multiple scans. (a) Sensor placement around objects/scenes and (b) rendering of original mesh.

reconstruction aims to create smooth surface descriptions from dense or sparse depth measurements that are typically corrupted by outliers and noise. Surface reconstruction of objects with arbitrary topology has usually been performed on triangular meshes because polygonal meshes provide a simple piece-wise planar surface representation that gives a first order approximation without constraints on topology (Hilton et al., 1996). In addition, a triangulated mesh can be easily constructed for a single range image. Existing integration algorithms aim to merge constrained triangulations of individual range images.

A typical surface reconstruction task consists of four steps: multi-view range data acquisition, view registration, 3D model building (view integration), and model optimization (Hilton et al., 1996). Multi-view data acquisition aims to eliminate occlusions inherent in single range images, and to provide redundant (overlapped) data sets. An example of multi-view scans and reconstructed 3D model of an office scene is shown in Fig. 2.2. View registration is to align multi-view data sets into a common coordinate system. The model building process, also called view integration, aims to create a single surface representation from the sample points contained in multiple registered data sets. Model optimization aims to adjust the quality of meshes including resolution, smoothness, etc. to a desired level. Major model optimization tasks include mesh simplification, mesh decimation, mesh smoothness, mesh denoising, etc. View integration algorithms are classified into five categories: (i) point-normal implicit surface (Hoppe and Derose, 1992), (ii) mesh implicit surface (Hilton et al., 1996), (iii) canonic views (Soucy and Laurendeau, 1995), (iv) mesh zippering (Turk and Levoy, 1994), and (v) mesh growing (Rutishauser et al., 1994).

Hoppe and Derose (Hoppe and Derose, 1992) proposed a general method for constructing an implicit surface representation from unstructured 3D points. Polygonal models were then generated using a marching cube approach (Lorenson and Cline, 1987). The algorithm is static

in the sense that all the image data are required prior to the polygonization process. This method is not specifically developed for range image integration, and does not reliably reconstruct the object topology. However, this method can be easily applied to range images. Hilton (Hilton et al., 1996) proposed a reliable view integration algorithm for complex objects based on a continuous implicit surface function. The surface function is constructed from step discontinuity constrained triangulations of individual range images. Soucy and Laurendeau (Soucy and Laurendeau, 1995) integrated range images using canonic subsets of the Venn diagram. Each of the canonic subsets represents the overlap between a subset of the 2.5D range images, and is associated with a 2D viewpoint reference frame. The 2D reference frames are used to eliminate redundant data and merge intersecting regions. They have extended their algorithm to be dynamic allowing the sequential integration of new range images. Turk and Levoy (Turk and Levoy, 1994) developed a dynamic mesh zippering algorithm. Overlapping regions of meshes are eroded and the boundary correspondence are found by operations in 3D space. A local 2D constrained triangulation is then used to join overlapping mesh boundaries to form a single mesh. Rutishauser et al. (Rutishauser et al., 1994) retriangulated two overlapping meshes using local constraints on triangle shapes in 3D space. Chen and Medioni (Chen and Medioni, 1992) used a triangle mesh which expands inside a sequence of range maps.

As middle-level primitives, the advantages of polygonal meshes lie in their abilities to represent arbitrary shapes with various levels of details by adjusting mesh resolutions. On the other hand, polygonal meshes have limitations since they are approximate and scale-dependent. Higher-level surface characterizations must be explicitly maintained with the mesh. The required resolution of the mesh may vary between applications. An active research topic is devoted to approaches coarsening dense meshes or refining coarse meshes according to application requirements. Another disadvantage of polygonal meshes is high computation and storage costs caused by a huge number of representation elements. For instance, approximately 3,000 vertices and 6,000 triangles are needed to represent a teapot of typical size with visually acceptable accuracy. Therefore, these surface representations, when expressed by a set of triangle or polygon meshes, can only be sufficient for applications that need simple geometric representation such as a visualization task. High-level tasks such as object recognition need more structural primitives to provide more compact information. In addition, tasks that require fast or real-time operations need more efficient primitives.

2.1.3 Generalized Cylinders

Generalized cylinders are the first dedicated part-level models in computer vision (Binford, 1971). Generalized cylinders have influenced much of the model-based vision research in the past two decades. As shown in Fig. 2.3, a generalized cylinder is formed from a volume by sweeping a two-dimensional set along an arbitrary space curve. The set may vary along the curve (axis). Therefore, definitions of the axis and the sweeping set are required to define a generalized cylinder. The axis can be defined as a function of arc length s in a fixed coordinate system (x, y, z)

$$a(s) = (x(s), y(s), z(s)). \tag{2.4}$$

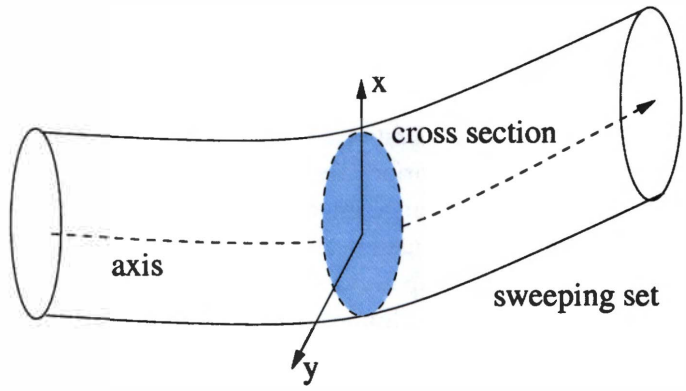


Figure 2.3: Plot of a generalized cylinder. A generalized cylinder is defined as a volume formed by sweeping a cross section along an axis (Binford, 1971).

The sweeping set is more conveniently expressed in a local coordinate system which is defined at the origin of each point of the axis $a(s)$. The sweeping set can be defined by a cross section boundary, which is parameterized by another parameter r

$$\text{sweeping set} = (x(r,s), y(r,s)). \tag{2.5}$$

This definition is very powerful and describes a large variety of shapes. To limit the complexity and simplify the recovery of generalized cylinders from images, constraints are often incorporated. Straight axes and constant sweeping sets are two main constraints added to generalized cylinders to represent regular shapes.

Generalized cylinders (GCs) are particularly attractive for representing elongated shapes where an axis is easy to define. In such cases, the axis of GCs often provides an intuitive method to conceptualize the design of an object, and a method of reliably recovering useful statistics about the shape of the object. On the other hand, shapes whose cross sections contain only one closed contour cannot easily be described by generalized cylinders when it is difficult to define an axis. It is possible to extend the primitives (GCs) to handle these objects, but it may not prove to be the most efficient representation of the objects with respect to design and recognition (Campbell and Flynn, 2001). An often cited early vision system which applied generalized cylinders is the ACRONYM system (Brooks, 1983). This is a model-based system for interpretation of airport scenes. Edges in a 2D image that form perceptually relevant entities and are defined as projections of generalized cylinders, are combined so that in the end a 3D interpretation of an intensity image is obtained. Models of airplanes that the system is to recognize are represented by generalized cylinders. Recovery of generalized cylinders from intensity images has been studied by many researchers. Especially notable for their work in this area are Rao and Nevatia (Rao and Nevatia, 1988), Mohan and Nevatia (Mohan and Nevatia, 1989), and Zerrough and Nevatia (Zerrough and Nevatia, 1999; Zerrough and Neviata, 1994). The recovery of generalized cylinders from intensity images seems to be overly complex since it must rely on complicated rules to group low-level image features, such as edges, corners, and surface

normals, into features of larger granularity (i.e., symmetrical contours or cross-sections), and to finally assemble them into generalized cylinders. These problems are due partly to the complicated parameterization of generalized cylinders, and to the lack of a fitting function that would provide a direct evaluation criteria on how well the model (GC) fits the image data (Jaklič et al., 2000). Consequently, only the recovery of a restricted subset of generalized cylinders such as straight homogeneous generalized cylinders (SHGCs) has been completed in the literature to date.

2.1.4 Geons

Geons, as qualitative primitives, consist of a set of solid blocks which are derived from generalized cylinders. They were proposed by Biederman (Biederman, 1985) in the context of human perception. The set of geons consists of 36 primitives obtained by changing axis shape, cross-section shape, cross-section sweeping function, and cross-section symmetry of generalized cylinders. The geons are depicted in Fig. 2.4. The 36 primitives of geons can be readily detected by an analysis of relatively perfect 2D line drawings. The putative component shapes are hypothesized to be simple and typically symmetrical, with no sharp concavities. The object components that can be differentiated on the basis of perceptual properties are sometimes difficult to detect, and are relatively independent of viewing position and degradation. Therefore, geons have been proposed as a basis for object recognition of 3D objects observed from a single 2D view. Geons are essentially conceptual or meta-models of parts which are normally implemented through generalized cylinders. They can also be expressed in terms of superquadrics. Several methods for qualitative, geon-type shape recovery based on superquadric models have been proposed (Dickinson and Pentland, 1992; Raja and Jain, 1992; Wu and Levine, 1993).

2.1.5 Superquadrics

Alan Barr (Barr, 1981) is the first to introduce superquadrics to computer graphics in early 1980s. Superquadrics appeared in computer vision as an answer to some of the problems with generalized cylinders (Pentland, 1986). As a subclass of generalized cylinders, superquadrics are a family of geometric solids which can be interpreted as a generalization of basic quadric surfaces and solids. A family of superquadrics is illustrated in Fig. 2.5. With only a few parameters, superquadrics can represent a large variety of standard geometric solids as well as smooth shapes, making superquadrics much more convenient for object representation. A superquadric is defined explicitly as

$$r(\eta,\omega) = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} a_1 \cos^{\varepsilon_1}\eta \cos^{\varepsilon_2}\omega \\ a_2 \cos^{\varepsilon_1}\eta \sin^{\varepsilon_2}\omega \\ a_3 \sin^{\varepsilon_1}\eta \end{bmatrix}, \tag{2.6}$$

where parameters a_1, a_2 and a_3 determine the size of the superquadric in the x, y and z directions, while ε_1 and ε_2 determine the shape. Superquadrics can represent a large set of shapes with various ε_1 and ε_2 as shown in Fig. 2.5. These shapes are convex only when $\varepsilon_1, \varepsilon_2 \in [0, 2]$.

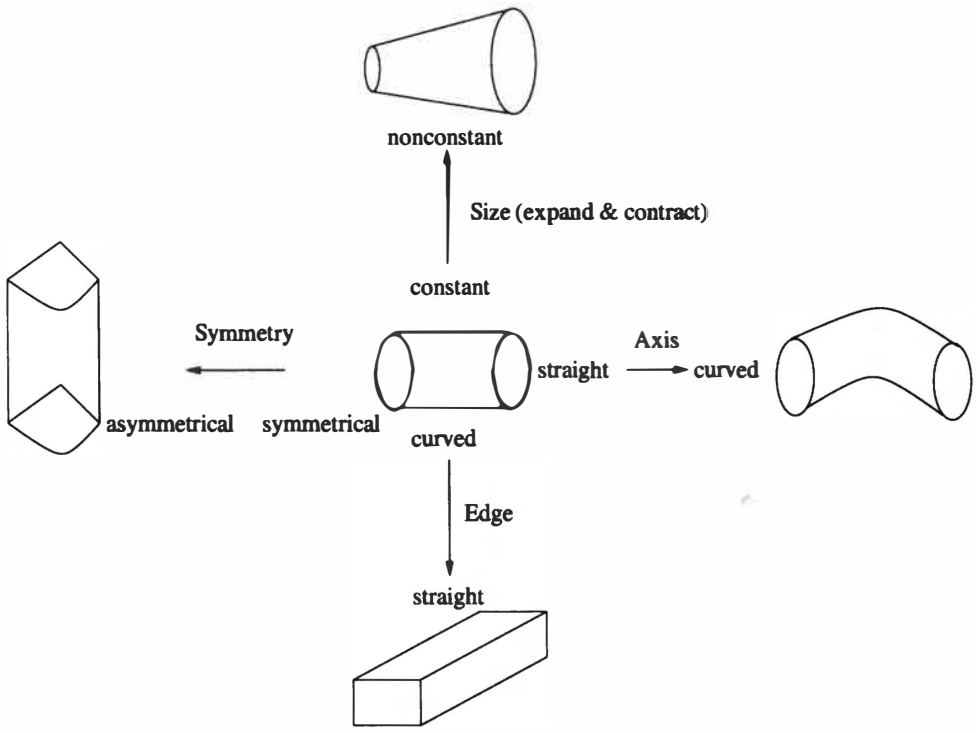


Figure 2.4: Shapes of geons. Geons consist of 36 primitives formed by varying size, symmetry, edge, and axis of a generalized cylinder.

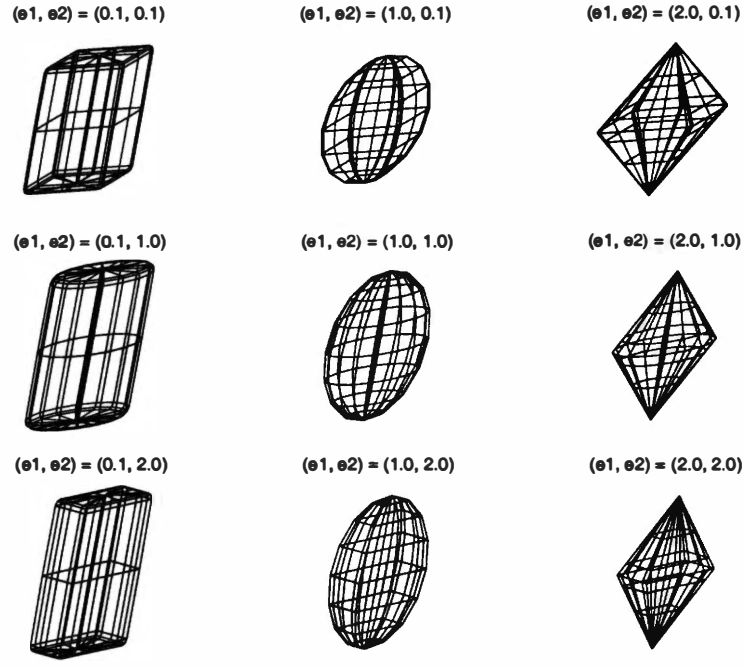
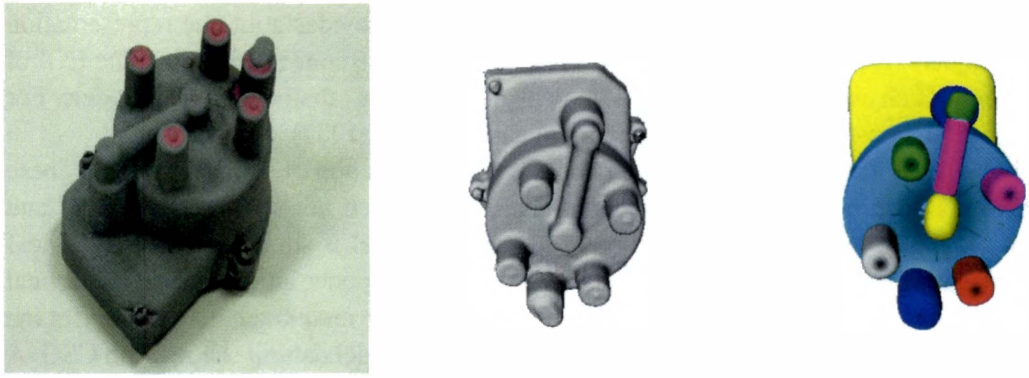


Figure 2.5: Superquadric shapes with various shape parameters (ε_1, ε_2).



(a) (b) (c)

Figure 2.6: Superquadric representation results for a distributor cap. This mesh is a reconstruction from multiple range images using the IVP Ranger System. The mesh consists of 58,975 vertices and 117,036 triangles while the decomposition consists of 13 parts. The original mesh is segmented using our part decomposition algorithm. Each of the decomposed parts is fitted to a superquadric model. Each superquadric is rendered in 3D using quad meshes based on recovered superquadric parameters. (a) Photograph of original object, (b) rendering of original mesh, and (c) rendering of recovered superquadrics.

As one of the volumetric primitives, superquadrics are the most compact and efficient representation primitives. Moreover, superquadrics can be deformed globally and locally by stretching, bending, tapering or twisting, and then be combined using Boolean operations to build more complicated objects. Therefore, an object with complex geometries can be represented by superquadrics for each of its single parts. For instance, a distributor cap and its superquadric representation are shown in Fig. 2.6. As demonstrated in this figure, although the overall geometry of the distributor cap is complex, and cannot be represented by a superquadric, each single part of the distributor cap is a regular shape, and can be successfully represented by a superquadric. One of the most attractive characteristics of superquadrics is their interchangeable implicit and explicit defining functions. The implicit definition is differentiable everywhere, and is especially suitable for model recovery. On the other hand, the explicit form is convenient for visualization/rendering. A more detailed survey of superquadric representation algorithms is presented in Section 2.2.

2.1.6 Other Volumetric Primitives

In addition to the primitives discussed previously, there are several other volumetric primitives including Gaussian images, extended Gaussian images (EGI), Constructive Solid Geometry (CSG), blob models, hyperquadrics, and higher order polynomials. These additional primitives attempt to represent an object as an entity in its own coordinate system. Gaussian images and EGIs are mappings of surface normals of an object onto the well-known Gaussian Sphere. Each point on the Gaussian Sphere corresponds to a surface normal of the object. An EGI is an

abstract model which represents a class of objects. The EGI provides a unique representation of convex objects (Horn, 1988), but it may correspond to infinite concave objects. The EGI of simple objects, such as cylinders or spheres, may be more complex than other solid models. For irregular objects, however, EGI is generated more easily (Shirai, 1987).

The basic idea of CSG is to construct 3D bodies from a selection of solid primitives. These primitives include cuboids, cylinders, spheres, and cones which are scaled, positioned, and combined by union, intersection, and difference. A CSG model is stored as a tree, with leaf nodes representing the primitive solid, and edges enforcing precedence among the set theoretical operations. CSG models define properties such as object volume unambiguously, but suffer the drawback of being non-unique. Furthermore, it is not easy to model natural shapes with CSG. A more serious drawback is that surface recovery is not straightforward from a CSG description. Such a procedure is computationally very expensive (Sonka et al., 1998).

Blob models of complex shapes can be recovered by minimizing an energy function and recursively splitting field primitives (Muraki, 1991b). As with CSG, this method is computationally very expensive. Hyperquadrics are a generalization of superquadrics which offer a larger shape flexibility through a larger number of parameters (Hanson, 1988). A given hyperquadric shape, however, can have more than one set of parameters (i.e., there is a many-to-one relationship between parameter sets and hyperquadric shapes). Higher-order polynomials can represent similar shapes as superquadrics, and are mathematically easier to manipulate. Fitting of implicit fourth order polynomials is discussed in (Tasdizen and Tarel, 1999; Taubin, 1991). A survey of free-form object representation and recognition techniques is provided in (Campbell and Flynn, 2001).

2.2 Superquadric Representation Algorithms

A typical superquadric representation method consists of input data, the recovery strategy, and the measurement of how well the recovered superquadrics fit the input data, and the complexity of objects which can be handled. Superquadric representation methods can be classified into different categories according to (i) data type (range, stereo, and intensity image), (ii) recovery strategy (analytic, minimization of an objective function, and point distribution model), and (iii) complexity of objects (single- or multi-part objects). Therefore, in this section, we review superquadric representation algorithms in these aspects including types of data used, recovery strategies, and complexities of objects and scenes to be represented.

2.2.1 Different Types of Input Data

Various types of images including intensity images, stereo images, and range images have been used as input in superquadric representation methods. The most appropriate type of images are range images and others containing dense and explicit 3D information of objects, such as images obtained by modern medical imaging techniques. The representation is also possible using pairs of stereo intensity images. The drawback of stereo is a sparse and non-uniform distribution of 3D points. Recovering superquadrics from single intensity images is more difficult.

Range image. Range images have been the most commonly used type of data for superquadric representation (Gross and Boult, 1988; Hager, 1994; Han et al., 1993; Pentland, 1986; Solina and Bajcsy, 1990). The underlying depth information of range images directly fits the images to superquadrics.

Stereo data. Metaxas (Chan and Metaxas, 1994; Metaxas and Dickinson, 1993) studied the problem of superquadric recovery from stereo and multiple views in the framework of qualitative shape recovery (Dickinson and Pentland, 1992) as well as physics-based recovery techniques (Metaxas and Terzopoulos, 1993). Stereo images have lost their popularity in superquadric representation due to their own drawbacks and the decreasing cost of range scanners.

Single intensity image. Intensity images are normally used for derivation of contours, edges or silhouettes which serve as low-level features in superquadric recovery (Pentland and Sclaroff, 1991; Vidmar and Solina, 1992). Vidmar and Solina (Vidmar and Solina, 1992) proposed a two-step method for the recovery of superquadrics from 2D contours. First, the position of the superquadric is determined. Next, the shape and orientation of the superquadric are reconstructed using an iterative least-square method. The contour points should lie on the surface of the superquadric model. For a specific contour, several possible superquadric models are usually derived. A better solution can be derived by using additional information, such as a few range points.

Dickinson (Dickinson and Metaxas, 1992; Dickinson and Pentland, 1992) proposed a method based on distributed aspect matching for superquadric recovery from contours, which were extracted from intensity images. The approach first took a set of 3D volumetric modeling primitives and generated a hierarchical aspect representation based on the projected surfaces of the primitives. This aspect graph was then matched with the segmented intensity image. Obtained matching probability can be used in object recognition. Pentland (Pentland, 1987) proposed a preliminary algorithm for recovering superquadrics from intensity images of single-part objects. Due to the utilization of intensity images, this algorithm was limited to shape recovery of single-part objects and failed to provide complete 3D information on position and orientation of the superquadric. To improve this method, Pentland and Sclaroff (Pentland and Sclaroff, 1991) proposed an algorithm in which a set of 2D silhouettes of a superquadric of different shapes, sizes, and orientations was used to generate a set of hypothesis about corresponding models in images. This approach is computationally less expensive than the previous work in (Pentland, 1987).

2.2.2 Different Recovery Strategies

Superquadric recovery methods can be classified according to recovery strategies into analytical methods, objective function minimization methods, and point distribution model methods. Despite different recovery strategies, the goal of superquadric representation methods is to find a superquadric that best fits a given image. Most superquadric representation approaches define an objective function, and obtain superquadric parameters through minimizing the objective function.

Analytical methods. Pentland (Pentland, 1986) presented an analytical solution for superquadric recovery. This approach utilized the dual relationship between superquadric surfaces and surface normals to derive superquadric parameters. Nevertheless, this approach demonstrates limited success with only simple synthetic images.

Objective function minimization methods. Pentland (Pentland, 1986) used a coarse grain exhaustive search in the space of superquadric parameters to obtain a set of the best goodness-of-fit hypotheses of individual superquadric models presented in a range image. This method is computationally very expensive due to the extensive initial search and generation of range images to compute goodness-of-fit. The problem of direct recovery of individual superquadrics from range images was solved by Solina and Bajcsy (Solina and Bajcsy, 1990). They presented an error metric based on the inside-outside function of the definition of superquadrics. To compensate the incomplete information in a single view and solve the ambiguity, the minimum volume constraint was added to the objective function. The Levenberg-Marquardt algorithm, which is a standard solution to the non-linear least-square minimization problem, was used to minimize the objective function.

Gross and Boult (Gross and Boult, 1988) followed a similar strategy, but studied different objective functions. In particular, they stated that the measure of fit based on radial Euclidean distance was a better metric than the inside-outside function used by Solina and Bajcsy (Solina and Bajcsy, 1990). However, the difference in recovered superquadrics could be visually negligible as stated in (Ferrie et al., 1993). The inside-outside function has been more commonly used due to its lower computation cost. The same error metric proposed by Solina and Bajcsy (Solina and Bajcsy, 1990) adjusted for tapering and bending deformations was successfully used to represent deformed objects (Solina and Bajcsy, 1990). Whaite and Ferrie (Whaite and Ferrie, 1991) proposed improvements on the error metric to reduce its bias.

Ferrie et al. (Ferrie et al., 1993) proposed a bottom-up strategy based on sequential application of different techniques to extract image regions corresponding to convex volumetric parts. They used Darboux frames to describe object surfaces and snake contour models to interpolate between image features that partition the object surfaces into their constituent parts. Finally, they fit superquadrics to individual parts. The experimental results are limited to cases where each segmented surface patch corresponds to a single volumetric model. This is a major drawback of all bottom-up approaches which do not use models to guide the segmentation process. A simple object, like an L-shaped object, will fail to be represented by this method.

Gupta and Bajcsy (Gupta and Bajcsy, 1992; Gupta and Bajcsy, 1993) used biquadric surface patches to pre-segment a range image. To solve the problem of multiple surfaces belonging to a single superquadric model, they grouped surfaces along convex discontinuities of surface normals. Leonardis et al. (Leonardis et al., 1995; Leonardis et al., 1997) presented a recover-and-select paradigm to recover superquadric models of articulated objects from unsegmented range images. The recover-and-select paradigm consists of two interleaving stages: model recovery and model selection. The main advantage of this classify-and-fit approach is that the performance of fitting is constantly monitored. Jaklič (Jaklič, 1997) extended this approach to recover superquadric models from multi-view range images. The close-form mathematical expression for inertial moments of superquadrics was derived in this work. An original approach

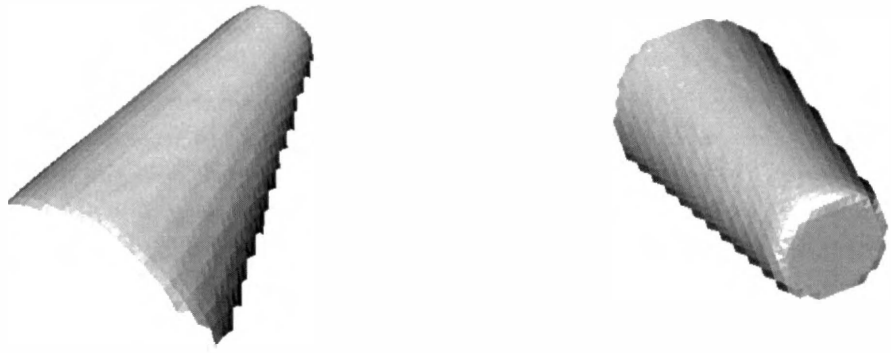
was presented to estimate the rigid transform between superquadric representations obtained from two different views.

Hu and Wee (Hu and Wee, 1995) proposed a robust 3D part extraction method which utilized an adaptive weighted partial data minimization technique. Demonstrating that the objective function defined by Solina and Bajcsy (Solina and Bajcsy, 1990) was sensitive to outliers and not robust, they modified the error metric by including weighted parameters. With the same objective function defined by Solina and Bajcsy, Zha et al. (Zha et al., 1998) proposed a recursive fitting-and-splitting algorithm for modeling objects with superquadrics. Zha et al. (Zha et al., 1998) extracted a dividing plane from range data to partition the original data set into two disjoint subsets, which were further treated respectively. However, this method failed to provide unique representations for some smoothly curved objects.

Zhou and Kambhamettu (Zhou and Kambhamettu, 1999) extended superquadrics by introducing exponential functions to represent more complicated non-symmetric objects. They modified exponents in the definition of superquadrics and the objective function from constants to exponential functions of latitude and longitude angles, which are expressed in the object-centered spherical coordinate system. Van Dop and Regtien (Dop and Regitien, 1998) compared four different objective functions for superquadric representation. They concluded that the objective function proposed by Solina and Bajcsy (Solina and Bajcsy, 1990) tended to recover square-shaped superquadrics and failed to deal with data containing outliers. The objective function with robustness and background constraints demonstrated the best performance on superquadric recovery in the presence of outliers in data sets. However, only a single cylindrical object was tested in this work and more comprehensive experiments with more thorough analysis are needed to support the statement.

Terzopoulos and Metaxas (Terzopoulos and Metaxas, 1991) formulated deformable superquadrics which incorporate both global deformations representing prominent shape features and local deformations capturing surface details. Though locally deformed models are flexible to represent nearly free-form objects, the implementation is very complicated. Compared with local deformations, the global deformations are more tightly integrated with superquadrics, and can be easily implemented. Bardinet et al. (Bardinet et al., 1998) presented a method using superquadric models and free-form deformations to represent unstructured 3D medical data. Although free-form deformations are more like global deformations in the sense of definitions, they can represent local details on surfaces to a certain level as local deformations.

Point distribution model methods. The point distribution model (PDM) is a statistical finite element model which is built from a training set of labeled contour landmarks of a large number of shapes. Pilu et al. (Pilu et al., 1996) proposed a method for recovery of deformed superellipses based on the PDM which can also be extended to superquadrics. Although PDMs are normally trained on actual shapes, Pilu et al. trained them on a large number of randomly generated deformed superellipses. For PDM training, the points must be labeled and put in correspondences across the whole training set. The mean shape is calculated by averaging each coordinate point, and the covariance matrix of the points is computed. The eigenvalue decomposition of the covariance matrix enables the approximation of shapes in the training set by using a weighted sum of the most significant eigenvectors. The weights are called modes of



(a) (b)

Figure 2.7: Synthetic range data of a tapered cylinder. Scans from two views are acquired.

variations. Pilu et al. used the first seven modes to represent deformed superellipses. Among the seven modes, the first four modes are highly correlated to the sizes (a_1, a_2) and to the bending and tapering factors. However, the shape parameter ε is not strongly correlated to a particular mode since ε does not affect major structural change of the superellipsoid.

2.2.3 Objects in Various Complexities

Most early research on superquadric representation concentrated on representing single-part objects from single-view intensity or range images (Dop and Regitien, 1998; Gross and Boult, 1988; Gupta and Bajcsy, 1992; Hager, 1994; Han et al., 1993; Pentland, 1986; Solina and Bajcsy, 1990; Zhou and Kambhamettu, 1999) by assuming that the images have been pre-segmented into single-part objects. This category of research is only concerned with the data fitting process including objective functions, fitting measurements, convergence speed, etc. The volumetric nature of superquadrics enables correct models to be recovered from a single-view image captured from a degenerate viewpoint (Jaklič et al., 2000; Solina and Bajcsy, 1990), which is a significant advantage over surface primitives.

The essential weakness of existing superquadric representation methods from single-view images is that confidence and accuracy of recovered superquadrics tend to be low due to incomplete information (partial surface) contained in single-view images. In addition, recovered superquadric parameters are subject to the viewpoint from which the scan is taken. Differences in superquadric parameters recovered from various viewpoints are not negligible in many cases. For instance, superquadric parameters recovered for the object shown in Fig. 2.7 from two views tend to be different due to different surfaces scanned. How this issue can be addressed remains unknown in the literature.

For complex, multi-part objects or multi-object scenes, there are three major types of superquadric representation approaches. The first type of methods incorporates an image segmentation step priori to superquadric representation (Ferrie et al., 1993; Gupta and Bajcsy, 1992; Gupta and Bajcsy, 1993; Hu and Wee, 1995; Metaxas and Dickinson, 1993; Metaxas and Terzopoulos, 1993; Zha et al., 1998). The image segmentation step involved in these methods is either inefficient or unreliable since superquadrics can only be successfully recovered from volumetric parts rather than regions. For example, Ferrie et al. (Ferrie et al., 1993) proposed a bottom-up strategy based on sequential application of different techniques to extract image regions corresponding to convex volumetric parts. They used Darboux frames to describe object surfaces and snake contour models to interpolate between image features that partition the object surfaces into their constituent parts. Finally, they fit superquadrics to individual parts. The experimental results are limited to cases where each segmented surface patch corresponds to a single volumetric model. Gupta and Bajcsy (Gupta and Bajcsy, 1992; Gupta and Bajcsy, 1993) used biquadric surface patches to pre-segment a range image. To solve the problem of multiple surfaces belonging to a single superquadric model, they grouped surfaces along convex discontinuities of surface normals. In summary, the segmentation step included in these superquadric representation methods segments objects into regions (surface patches) instead of volumetric parts, which can truly reflect the part-based nature of superquadrics.

The second type of method directly recovers superquadrics from single-view range images without pre-segmentation (Jaklič et al., 2000; Leonardis et al., 1997). Compared to the superquadric representation approaches for single-part objects, these methods suffer more from occlusions caused by single-view images due to the complexity of objects. In addition, this type of method has high computational costs due to a large number of iterations required to converge to the optimal superquadric representation. The final type of methods used local deformations to represent complex objects, mostly biomedical organs (Bardinet et al., 1998; Terzopoulos and Metaxas, 1991). This type of method suffers both high computational costs and difficult selection of initial models during the deformation process. For this reason, these methods involving local deformations have been primarily applied to nonrigid objects such as biomedical organs, for which local deformations are essential. Compared with the early work on superquadric representation of single-part objects, these types of methods can represent complex objects and scenes and have more practical applications in tasks such as robotic navigation, object recognition, etc. These three types of superquadric representation approaches have their own advantages and limitations. Which approach should be used heavily depends on the application involved.

In summary, there are two major drawbacks for existing superquadric representation methods of multi-part objects. The first is a lack of an efficient part decomposition method, and the other is that only single-view images are used for superquadric recovery. For instance, no existing superquadric representation method can handle the multi-part object and the multi-object scenes shown in Fig. 2.8. For these objects, it is too difficult to choose an optimal viewpoint from which each part is visible due to severe occlusions. Therefore, existing superquadric representation methods utilizing single-view range images yield incomplete and incorrect superquadrics. In addition, it is insufficient to recover superquadrics from these objects without pre-segmentation due to the complexity of the objects.

(a) (b)

Figure 2.8: Multi-part object and multi-object scene. (a) Photograph of a distributor cap (multi-part object) and (b) photograph of a scene containing multiple objects simulating a bin picking task.

2.3 Part Decomposition

As stated in Section 2.2.3, part decomposition can significantly benefit superquadric representation of multi-part objects. When multi-part objects or multi-object scenes are present as shown in Fig. 2.8, part decomposition is indispensable to correct superquadric representation. However, the segmentation step included in the existing superquadric representation methods segments objects or scenes into regions (surface patches) instead of volumetric parts, which can truly reflect the part-based volumetric nature of superquadrics.

Besides superquadric representation, many other tasks in computer vision, computer graphics, and reverse engineering are directly performed on objects (models). Those object-centered algorithms become more difficult when the treated object is complicated, for example, when it contains multiple parts. Part decomposition is a crucial step in such applications since it can simplify the original problem for multi-part, complex objects into several subproblems, each dealing with their constituent single, simpler parts. In applications such as object recognition, shape description and representation, and object manipulation, part decomposition is an indispensable pre-processing step, and can further reduce the efforts involved with the original multi-part objects (Hoffman and Richards, 1984; Pentland, 1981). Different from the image segmentation concept, in which regions (surface patches) are segmented from either edge- or region-based methods (Baccar et al., 1996; Burgiss et al., 1998; Hoover et al., 1996; Jiang, 2000; Zhang et al., 2000), part decomposition aims to segment compound objects into their constituent parts.

Part decomposition classifies data points on object surfaces into groups, each of which corresponds to one physical part of the object. For example, an object composed of two articulated parts, e.g., a cube and a cylinder, will be segmented into six planar surfaces for the cube, and two planar surfaces and a circular surface for the cylinder from a region segmentation algorithm. Instead, a part decomposition algorithm decomposes the object into its constituent parts which

(a) (b)

Figure 2.9: Part decomposition results for a mug. This mesh is a reconstruction from Hughes Hoppe at Microsoft Research. The mesh consists of 1,725 vertices and 3,450 triangles while the decomposition consists of two parts. (a) Rendering of original mesh and (b) decomposed parts labeled in different colors.

include a cube and a cylinder. Obviously, the segmentation results from part decomposition can better facilitate superquadric representation of the objects. An example of part decomposition of a mug is shown in Fig. 2.9. Compared with image segmentation, part decomposition results in higher-level segmented components and is more appropriate for high-level, object-centered tasks such as object representation and recognition (Ferrie et al., 1993).

The part theory can be dated back to the *transversality regularity* proposed by Guillemin and Pollack (Guillemin and Pollack, 1974) as a human perception theory. The transversality regularity is defined in the following. When two arbitrarily shaped surfaces are made to interpenetrate they always meet in a contour of concave discontinuity of their tangent planes. Fig. 2.10 shows the transversality regularity in 2D space. Notice that the silhouette of the articulated shape is not smooth at the two points, point A and B, where the silhouette of one of its parts intersects the silhouette of the other part. At these two points, the direction of the silhouette's outline (i.e., its tangent direction) changes abruptly, creating a concave cusp at each of these two points. In 3D space, such concave discontinuities arise at every point on the surface of the composite shape where the two parts meet each other.

As suggested by Hoffman and Richards (Hoffman and Richards, 1984), part decomposition methods can be classified into primitive- or boundary-based. A primitive-based approach defines parts by their shapes, not by their contours of intersections. A boundary-based approach defines parts by their contours of intersection, not by their shapes. In particular, primitive-based approaches decompose objects into parts by measuring the shape similarity between image data and predefined part models. This type of approach can also be called object representation because the segmented parts are finally represented by shape primitives. The primitives that

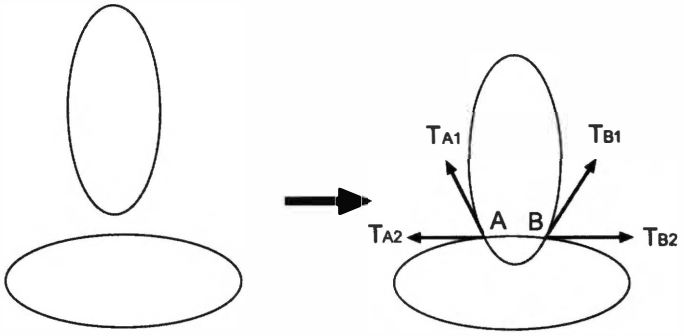


Figure 2.10: An illustration of the transversality regularity in 2D. Two arbitrary shapes always interpenetrate in a contour of concave discontinuity of their tangents. For each of the intersection points A and B, there are two discontinuous tangents T_1 and T_2 for the two intersecting curves.

have been used in part decomposition or object representation include spheres (Mohr and Bajcsy, 1983; O'Rourke and Badler, 1979), polyhedra (Hoppe and Derose, 1992; Hilton et al., 1996), generalized cylinders (Binford, 1971; Zerrough and Nevatia, 1999; Zerrough and Neviata, 1994), geons (Biederman, 1985; Raja and Jain, 1992; Wu and Levine, 1993), and quadrics and superquadrics (Gupta and Bajcsy, 1992; Jaklič et al., 2000; Leonardis et al., 1997).

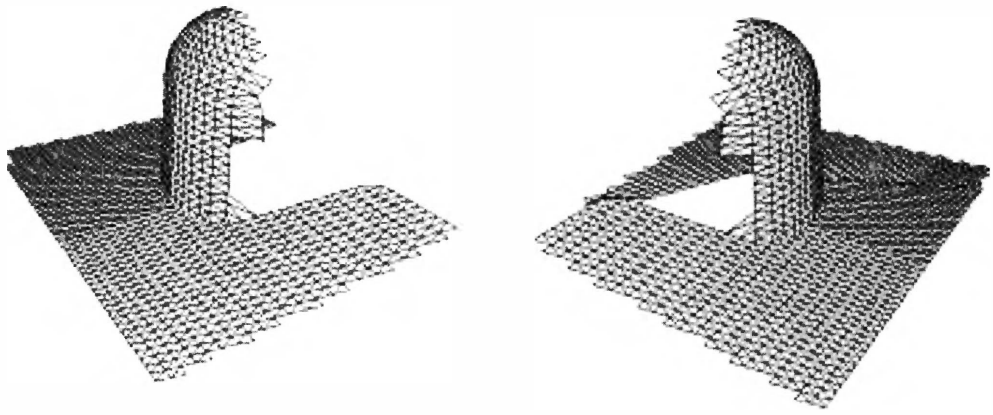
Despite the various primitives used, these approaches first hypothesize an object configuration composed of part models, assuming that the shape of each part is very similar to that of a particular model. Second, they evaluate a measure of similarity between the hypothesis and the true object shape. Dickinson et al. (Dickinson et al., 1997; Metaxas and Dickinson, 1993) used an aspect hierarchy of part shapes in a large set of distinctive model database. Surface patches were first identified using region growing or edge detection and then grouped into a potential part. These primitive-based part decomposition methods are advantageous when each part of an object can be represented by a primitive. In such cases, parts are not only segmented but represented in primitives. On the other hand, problems arise when the parts are not consistent with the available primitives. For instance, superquadrics can represent animal limbs, but are obviously inappropriate for faces, cars, shoes, etc. In such cases, nonunique or incorrect part segmentation occurs. In summary, primitive-based part segmentation approaches have limited versatility.

On the other hand, a boundary-based part segmentation approach, if its rule uses only geometry of surfaces, can apply to many objects whose bounding surfaces are amenable to the tools of differential geometry, which is not a severe restriction (Hoffman and Richards, 1984). Compared to the primitive-based approaches, the boundary-based part segmentation approaches can segment objects into parts without knowing in advance what the parts look like. This makes more sense because part segmentation is essentially a grouping process, which does not require a high-level primitive representation. In summary, boundary-based part segmentation approaches have greater versatility than primitive-based ones (Hoffman and Richards, 1984) in the sense of segmentation.

Two specific rules have been proposed for boundary-based part segmentation based on the theory of transversality regularity. Koenderink and Van Doorn (Koenderink and Doorn, 1982) presented parabolic lines as part boundary. At a parabolic line (do Carmo, 1976), one of the principal curvatures of the surface changes from convex (> 0) to concave (< 0). Hoffman and Richards (Hoffman and Richards, 1984) presented the *minima rule* to partition a surface into parts along contours of concave discontinuity of the tangent plane. The minima rule (Hoffman and Richards, 1984) divides a surface into parts at loci of negative minima of each principal curvature along its associated family of lines of curvature. Following the rule proposed by Koenderink and Van Doorn (Koenderink and Doorn, 1982), Rom and Medioni (Rom and Medioni, 1994) proposed a framework consisting of decomposing compound 3D objects into single parts and then describing those parts by generalized cylinders. Wu and Levine (Wu and Levine, 1997) presented a physics-based part segmentation approach. The novelty of this method is that part boundaries were detected by electrical charges instead of traditional curvatures for each vertex. Although the author claimed that this method was robust to mesh resolution and had several advantages over curvature-based part segmentation methods, the high computational cost involved in electrical charge evaluation prevented this method from being widely used.

Compared to the significant amount of research devoted to 2D intensity or range image segmentation over the last two decades (Bennamoun, 1994; Hoffman and Richards, 1984; Koara et al., 2000; Kumar and Goldgof, 1995; Leonardis et al., 1997; Pentland, 1981), 3D mesh segmentation is a relatively new research topic. Unlike part decomposition, mesh segmentation segments meshes into regions. Falcidieno and Spagnuolo (Falcidieno and Spagnuolo, 1992) segmented a mesh into similar curvature regions of concave, convex, planar, and saddle patches. Other mesh segmentation algorithms have been proposed to segment meshes into either planar surfaces (Papaioannou et al., 2000; Sacchi et al., 1999) or arbitrary patches (Mangan and Whitaker, 1999).

Differential parameters such as normals, principle curvatures, and principle directions are very useful in a wide variety of tasks including segmentation, surface classification, and surface reconstruction and registration. As suggested by (Krsek et al., 1998), curvature estimation methods can be classified into (i) approximation by an analytic surface, (ii) approximation by curves, (iii) convolution operators, and (iv) discrete curvature methods. Approximation by an analytic surface is a commonly used approach to estimate differential parameters. With this type of approach, an analytic surface is first locally fitted to input points, and the curvatures are computed for this approximate surface using the classical technique in (Besl, 1988). The methods in this category are further classified into linear (Hamann, 1993; Stokely and Wu, 1992) and nonlinear approximations (Sander and Zucker, 1990) according to the approaches used to fit surfaces with input data. Approximation of curves lying in the planar part of a surface can achieve faster estimation of differential parameters. Such methods first estimate curvatures for the curve and then calculate curvature for the surface that consists of the curves (Martin, 1998). Convolution operators combine computation of derivatives with noise filters (Thirion, 1996) and are suitable for depth maps based on regular grids. These methods are usually used to detect structures such as edges, peaks, etc. Discrete curvature methods estimate principal curvatures directly from triangulated surfaces utilizing polyhedral metrics (Besl, 1988;



(a) (b)

Figure 2.11: Self-occlusions occurred in two-view scans of an object.

Lin and Perry, 1982). These methods are quick and robust but are not able to estimate principal directions.

2.4 View Registration

Reconstructing 3D models of real-world objects or scenes from multi-view range images has gained tremendous attention over the last decade due to increasing research interests in 3D computer vision as well as the high accuracy of up-to-date range scanners with decreasing costs. Reconstructed 3D models are of greatest interest in applications such as robot navigation, object recognition, CAD modeling, computer graphics, and reverse engineering. Compared with stereo-based 3D reconstruction techniques, range-based methods, in general, provide higher accuracy and robustness. A typical 3D surface reconstruction process consists of three steps: the *data acquisition* step in which multi-view range data are captured from appropriately planned viewpoints, the *view registration* step in which the data obtained from various views are aligned into a common global coordinate system, and the *data integration* step in which overlaps are removed, and the geometry of the aligned data sets is optimized to a single mesh.

Multi-view data acquisition is essential since most 3D scanning devices can only capture partial surfaces of objects at a time due to the limitations in the physical design and the specific technology of the scanner. An example of partial scanning of an object is shown in Fig. 2.11. As observed from this figure, only part of the object surface is captured from a single viewpoint due to self-occlusions. Multi-view data acquisition is essential in order to construct a complete 3D model of an object. Since each scan is generally represented in an independent sensor coordinate system, the single-view data must be transformed from its own sensor coordinate system to a global object-centered coordinate system. This process is termed *view registration*, and received sustained attention in the research literature. The view registration is defined to

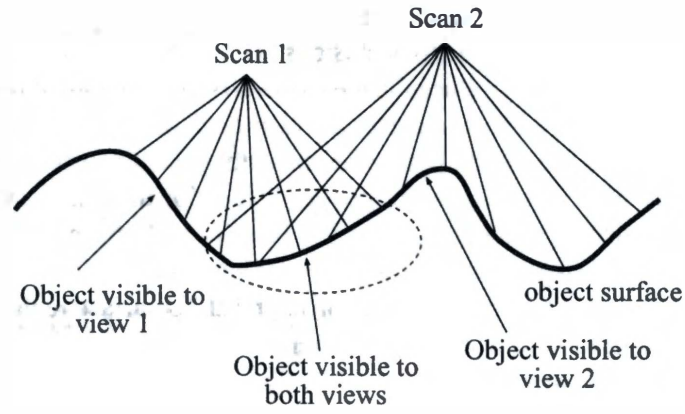


Figure 2.12: Scans from two views. Overlapping object surfaces scanned from two views need to be registered in order to be integrated into more complete data.

find a rigid transformation $T_i^j(p)$ between coordinate system i and j. The transformation $T_i^j(p)$ is often evaluated by $T_i^j(p) = Rp+d$, where R is a rotation matrix, and d is a translation vector. Variable p denotes a point represented by its (x, y, z) coordinates. In general, the transformation is obtained from overlapping regions as shown in Fig. 2.12.

View registration is crucial since it significantly affects the performance of the subsequent data integration step. A typical view registration approach consists of two stages: (i) coarse registration (initial estimation) of the transformation between views and (ii) refinement of the initial registration.

Coarse Registration. The goal of a coarse view registration is to compute a transformation that can approximately align two data sets acquired from two different views. The transformation needs to align the data close enough for a subsequent "fine registration" step. The coarse registration aims to improve the convergence of the fine registration to the global optimal solution. The coarse registration methods can be classified into manual and automatic approaches according to how corresponding points contained in different views are identified. Two types of manual registration approaches are possible. The first type uses a controlled scanning environment, where the transformations between the views' coordinate frames are obtained from the calibration of the scanner with the mechanics of the sensing environment. The problems with such scanning systems are objects need to be placed in precise locations in the scene. In addition, the scanners must be precisely calibrated. These constraints result in more expensive equipment and limitations on the types of objects that can be scanned. The other type of manual approach uses human interaction to obtain coarse registration between views. In such cases, human operators pick up corresponding points from different views, and transformations between these views can be evaluated from the correspondences. In summary, disadvantages of the manual registration approaches include (i) the mechanical device required for view information is

often unavailable or infeasible for range scanners, and (ii) the manually chosen correspondences tend to be inaccurate for range data due to occlusions.

For this reason, many efforts have been made to identify correspondent point pairs between views automatically. Once the correspondences are identified, the rigid transformation between the views can be easily obtained by computing the absolute orientation using orthonormal matrices (Horn et al., 1988). This automatic correspondence identification process is termed surface matching. In general, surface matching approaches identify and match similar points through encoding surface geometry (representation primitives) around points of interest.

Despite various representation primitives involved, most 3D surface registration approaches fall into a common framework, in which representation primitives are estimated for each point, and correspondent points between the views are identified by matching the representation primitives. Chen and Medinoi (Stein and Medioni, 1992) used changes in surface orientation to match local surface patches. A novel measurement of the difference between two relative normal distributions was proposed to provide good matches between corresponding points. Chua and Jarvis (Chua and Jarvis, 1996; Chua and Jarvis, 1997) formulated a new representation primitive, the point signature, which encodes the minimum distance of points on a 3D contour to a reference plane. Johnson and Herbert (Johnson and Hebert, 1998; Johnson and Hebert, 1999) also employed point features for surface matching. They proposed spin images as 2D histograms of the surface locations around a point. The spin image is generated by rotating a cutting plane around the point using the normal as the rotation axis. Yamany and Farag (Yamany and Farag, 1999) modified the spin image and proposed a surface signature to match surfaces. Bergevin et al. (Bergevin et al., 1995) proposed a view registration algorithm based on matching properties of triangles generated from a hierarchical tessellation of object surfaces. Roth (Roth, 1999) first identified points of interest from an intensity image, and then matched associated triangulated range images. Simultaneous or global registration of three or more views have been proposed in (Bergevin et al., 1996; Blais and Levine, 1995; Dorai and Jain, 1997; Eggart et al., 1996; Pulli, 1999; Stoddart and Hilton, 1996). Other surface matching methods can be found in (Wang et al., 2000; Yahia et al., 2000; Zhang and Hebert, 1999).

In conclusion, low-level (points and contours) and middle-level primitives (triangle meshes) have been extensively applied to surface registration. As one of the highest-level representation primitives (volumetric primitives), superquadrics are able to represent many common shapes, including boxes, cylinders, spheres, ellipsoids, etc. using a small number of parameters (Barr, 1981; Jaklič et al., 2000). In addition, the mapping from an object to the corresponding recovered superquadric is one-to-one because the recovered superquadric parameters uniquely determine the size, position, and orientation of the object in the world coordinate system. These properties make superquadrics a very powerful primitive in pose estimation and other computer vision tasks. Jaklič et al. (Jaklič et al., 2000) proposed a range image registration algorithm based on recovered superquadrics to register range images captured from two views.

Fine registration. Fine registration refines transformations provided by the coarse registration process to a more precise alignment between views. A typical quality criterion is the distance between corresponding points in different views. In this category, The iterative closest point (ICP) algorithm (Besl and McKay, 1992) has been widely used. Zhang (Zhang, 1994)

improved the ICP algorithm to handle outliers and occlusions. The improved ICP algorithm demonstrated better tolerance for larger uncertainties in the initial guess of pose, outliers, and missing data while still managing to converge to the optimal solution. Chen and Medioni (Chen and Medioni, 1992) employed orientation information to iteratively refine the initial transformations. Other ICP-based algorithms either utilized color information (Johnson and Kang, 1999) or increased computational efficiency (Rusinkiewicz and Levoy, 2001; Feldmar and Ayache, 1996; Guest et al., 2001; Sharp et al., 2002; Trucco et al., 1999; Yamany et al., 1998)

2.5 Background on Superquadrics

Superquadrics are geometric shapes generalized from basic quadric surfaces and solids (Barr, 1981). With a fairly simple parameterization, they can represent a large variety of geometric shapes. This property makes superquadrics suitable for object representation and recognition. Additionally, superquadrics include superellipsoids, superhyperboloids of one and two pieces, and supertoroids (Barr, 1981). However, since superhyperboloids and supertoroids are not commonly used, superquadrics are often used to represent *superellipsoids*. This dissertation is based on this assumption. Definitions and characteristics of regular superquadrics are described in this section.

2.5.1 Regular Superquadrics

A superquadric is defined explicitly as

$$r(\eta,\omega) = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} a_1 \cos^{\varepsilon_1}\eta \cos^{\varepsilon_2}\omega \\ a_2 \cos^{\varepsilon_1}\eta \sin^{\varepsilon_2}\omega \\ a_3 \sin^{\varepsilon_1}\eta \end{bmatrix}, \tag{2.7}$$

where $-\pi/2 \leq \eta \leq \pi/2$, $-\pi \leq \omega < \pi$, parameters a_1, a_2 and a_3 determine the size of the superquadric in the x, y and z directions, while ε_1 and ε_2 determine the shape of the cross section. Superquadrics can represent a large set of shapes with various ε_1 and ε_2 as shown in Fig. 2.5. These shapes are convex only when $\varepsilon_1, \varepsilon_2 \in [0, 2]$. From the explicit definition, the equations are derived as

$$\begin{aligned} (\frac{x}{a_1})^{\frac{2}{\varepsilon_2}} &= cos^{\frac{2\varepsilon_1}{\varepsilon_2}}\eta cos^2\omega \\ (\frac{y}{a_2})^{\frac{2}{\varepsilon_2}} &= cos^{\frac{2\varepsilon_1}{\varepsilon_2}}\eta sin^2\omega. \end{aligned} \tag{2.8}$$

A superquadric can then be defined implicitly as

$$F(x,y,z) \equiv \left[(\frac{x}{a_1})^{\frac{2}{\varepsilon_2}} + (\frac{y}{a_2})^{\frac{2}{\varepsilon_2}}\right]^{\frac{\varepsilon_2}{\varepsilon_1}} + (\frac{z}{a_3})^{\frac{2}{\varepsilon_1}} = 1. \tag{2.9}$$

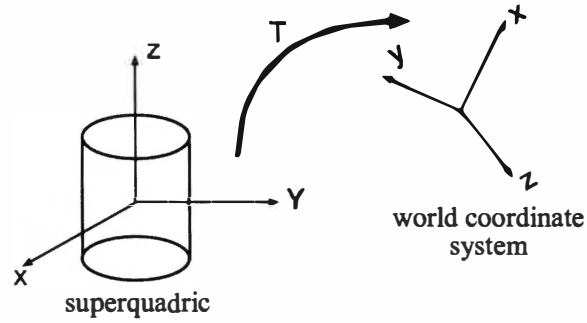


Figure 2.13: Homogeneous transformation for superquadrics. The transformation is between the canonical coordinate system of a superquadric and the world coordinate system.

Superquadrics have been presented only in the canonical coordinate system from the definitions. To express a superquadric in the world (global) coordinate system, a homogeneous transformation needs to be applied, which is often decomposed into rotation and translation. The relationship between the canonical coordinate system of a superquadric and the world coordinate system is depicted in Fig. 2.13. A superquadric is converted from its canonical coordinate system to the world coordinate system by

$$\begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} = T_t T_r \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix}, \tag{2.10}$$

where T, T_t, T_r denote the transformation, translation and rotation respectively, and

$$T_t = \begin{bmatrix} 1 & 0 & 0 & p_x \\ 0 & 1 & 0 & p_y \\ 0 & 0 & 1 & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{and} \quad T_r = \begin{bmatrix} n_x & o_x & a_x & 0 \\ n_y & o_y & a_y & 0 \\ n_z & o_z & a_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{2.11}$$

Therefore, the transformation and its inverse are

$$T = T_t T_r = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{2.12}$$

and

$$T^{-1} = \begin{bmatrix} n_x & n_y & n_z & -(p_x n_x + p_y n_y + p_z n_z) \\ o_x & o_y & o_z & -(p_x o_x + p_y o_y + p_z o_z) \\ a_x & a_y & a_z & -(p_x a_x + p_y a_y + p_z a_z) \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{2.13}$$

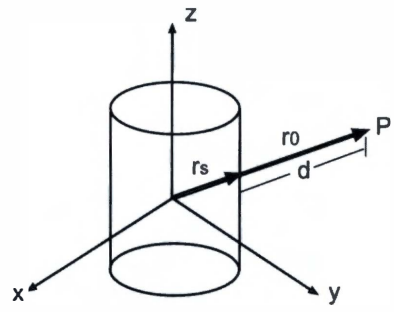


Figure 2.14: Radial Euclidean distance for superquadrics. The distance is between point P and a superquadric; d is the distance on the line through P and the center of the superquadric.

The rotation matrix can also be expressed with three independent Euler angles (ϕ, θ, ψ) as

$$T_r = \begin{bmatrix} C_\phi C_\theta C_\psi - S_\phi S_\psi & -C_\phi C_\theta S_\psi - S_\phi C_\psi & C_\phi S_\theta & 0 \\ S_\phi C_\theta C_\psi + C_\phi S_\psi & -S_\phi C_\theta S_\psi + C_\phi C_\psi & S_\phi S_\theta & 0 \\ -S_\theta C_\psi & S_\theta S_\psi & C_\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{2.14}$$

Therefore, the transformation can be expressed as

$$T = \begin{bmatrix} C_\phi C_\theta C_\psi - S_\phi S_\psi & -C_\phi C_\theta S_\psi - S_\phi C_\psi & C_\phi S_\theta & p_x \\ S_\phi C_\theta C_\psi + C_\phi S_\psi & -S_\phi C_\theta S_\psi + C_\phi C_\psi & S_\phi S_\theta & p_y \\ -S_\theta C_\psi & S_\theta S_\psi & C_\theta & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{2.15}$$

where C and S denote cos and sin functions respectively.

As a result, the implicit function of superquadrics in the world coordinate system has a total of 11 parameters (Solina and Bajcsy, 1990) expressed as

$$F(x_w, y_w, z_w) = F(x_w, y_w, z_w; a_1, a_2, a_3, \varepsilon_1, \varepsilon_2, \phi, \theta, \psi, p_x, p_y, p_z), \tag{2.16}$$

where parameters a_1, a_2 and a_3 define the size of superquadrics in the x, y and z directions, $\varepsilon_1, \varepsilon_2$ the shape, ϕ, θ, ψ the orientation, and p_x, p_y, p_z the position in the 3D space. We refer to the set of these parameters as $\Lambda_1 = \{a_1, a_2, a_3, \varepsilon_1, \varepsilon_2, \phi, \theta, \psi, p_x, p_y, p_z\}$.

Since the true Euclidean distance between a point and a superquadric is too complicated to calculate, the radial Euclidean distance is commonly used as a substitute. The radial Euclidean distance is defined as the distance from a point to a superquadric surface along the line through the point and the center of the superquadric. The radial Euclidean distance is depicted in Fig. 2.14, and can be evaluated based on the implicit definition of superquadrics. The radial Euclidean distance is derived as

$$d = |r_0| \left| 1 - F^{-\frac{\varepsilon_1}{2}}(x_0, y_0, z_0) \right| = |r_s| \left| F^{\frac{\varepsilon_1}{2}}(x_0, y_0, z_0) - 1 \right|, \tag{2.17}$$

where F is the implicit definition of superquadrics. Term (x_0, y_0, z_0) represents coordinates of point P in the canonical coordinate system. Variables r_0 and r_s are illustrated in Fig. 2.14. Consequently, the following properties hold

$$\begin{cases} F(x_0, y_0, z_0) = 1 & \Longleftrightarrow & \text{point is on the surface of the superquadric} \\ F(x_0, y_0, z_0) > 1 & \Longleftrightarrow & \text{point is outside of the superquadric} \\ F(x_0, y_0, z_0) < 1 & \Longleftrightarrow & \text{point is inside of the superquadric.} \end{cases} \tag{2.18}$$

Therefore, function F is also called an inside-outside function. The volume of a superquadric is derived from the area of a superellipse, and can be expressed as

$$V = 2a_1a_2a_3\varepsilon_1\varepsilon_2 B\left(\frac{\varepsilon_1}{2}+1, \varepsilon_1\right) B\left(\frac{\varepsilon_2}{2}, \frac{\varepsilon_2}{2}\right), \tag{2.19}$$

where the term $B(x, y)$ is a Beta function, and is defined by

$$B(x, y) = 2\int_0^{\pi/2} \sin^{2x-1}\phi \cos^{2y-1}\phi d\phi. \tag{2.20}$$

Similarly, the moments of inertia for superquadrics are derived from those of superellipses (Jaklič et al., 2000), which can be expressed in the following equations as

$$\begin{aligned} I_{xx} &= \frac{1}{2}a_1a_2a_3\varepsilon_1\varepsilon_2(a_2^2 B(\frac{3}{2}\varepsilon_2, \frac{1}{2}\varepsilon_2)B(\frac{1}{2}\varepsilon_1, 2\varepsilon_1+1) \\ &\quad +4a_3^2 B(\frac{1}{2}\varepsilon_2, \frac{1}{2}\varepsilon_2+1)B(\frac{3}{2}\varepsilon_1, \varepsilon_1+1)) \\ I_{yy} &= \frac{1}{2}a_1a_2a_3\varepsilon_1\varepsilon_2(a_1^2 B(\frac{3}{2}\varepsilon_2, \frac{1}{2}\varepsilon_2)B(\frac{1}{2}\varepsilon_1, 2\varepsilon_1+1) \\ &\quad +4a_3^2 B(\frac{1}{2}\varepsilon_2, \frac{1}{2}\varepsilon_2+1)B(\frac{3}{2}\varepsilon_1, \varepsilon_1+1)) \\ I_{zz} &= \frac{1}{2}a_1a_2a_3\varepsilon_1\varepsilon_2(a_1^2+a_2^2)B(\frac{3}{2}\varepsilon_2, \frac{1}{2}\varepsilon_2)B(\frac{1}{2}\varepsilon_1, 2\varepsilon_1+1). \end{aligned} \tag{2.21}$$

2.5.2 Globally Deformed Superquadrics

Research on deformable models has been very active in the areas of computer graphics, mechanical simulation, computer vision, and medical image processing. Due to the wide variety of man-made and natural objects, deformable models are frequently used to extend regular shapes. In applications involving visualizations, deformable models are used to design and create irregular shapes. For computer vision tasks focusing on representing existing objects with computer models, deformable models are used to represent objects which cannot be represented by regular primitives. Deformations can usually be classified into global and local deformations according to whether the deformation is performed globally or locally. Global deformations are able to deform models globally with low computation costs while local deformations can add local geometric details to the object surfaces at a much higher expense. In some cases, global and local deformations are combined to first roughly represent objects with globally deformed

models, and next in a refined manner with local deformations (Terzopoulos and Metaxas, 1991; Jaklič et al., 2000).

Global deformations used in computer graphics were first defined by Barr (Barr, 1984), in which global deformations were classified into scaling, tapering, bending, and twistering operators. These operators have been widely used by succedent researchers in computer graphics to create deformable models with or without variations (Liang and Wyvill, 2001; Liu and Sclaroff, 2001; Vemuri and Guo, 2000; Zhuang and Canny, 2000; Ramanathan and Metaxas, 2000). Solina and Bajcsy (Solina and Bajcsy, 1990) first introduced global deformations to superquadric representation in computer vision. They demonstrated that the original definitions of global deformations were more suitable for generating models in computer graphics fields, and had limitations when being used in reverse problems such as object representation tasks (Solina and Bajcsy, 1990; Jaklič et al., 2000). The major drawback with the global deformations defined by Barr is that they can only operate on a straight line. Furthermore, since the bending deformation is defined in a single plane, superquadrics to be recovered cannot rotate freely in order to align the bending plane with the raw data where bending is required (Jaklič et al., 2000). Realizing that global deformations must be defined in a manner that the object-centered coordinate system constrains the deformations as little as possible, Solina and Bajcsy (Solina and Bajcsy, 1990) modified tapering and bending deformations to be more suitable for model (superquadric) representation. The subsequent research on superquadric representation (Terzopoulos and Metaxas, 1991; Zhou and Kambhamettu, 2000; Gonzalez-Linares et al., 2000; Dickinson et al., 1997; O'Donnel et al., 1996; Kumar and Goldof, 1996; Kakadiaris et al., 1994; Chen and Medioni, 1994; Kervrann and Heitz, 1994; Park et al., 1994) has used global deformations defined by Solina and Bajcsy.

A shape deformation is a function D which explicitly transforms the coordinates of all surface points in 3D space based on deformation parameters (Solina and Bajcsy, 1990). This deformation process can be expressed as

$$P = D(p) = \begin{bmatrix} X(x,y,z) \\ Y(x,y,z) \\ Z(x,y,z) \end{bmatrix}, \tag{2.22}$$

where p represents a point (x, y, z) on the surface of a non-deformed solid, and P represents the corresponding point (X, Y, Z) after the deformation. Both variables p and P are expressed in the object-centered coordinate system. Any translation or rotation of the model must be performed after the deformation (Witkin et al., 1987) . This can be described schematically as

$$\text{Translation(Rotation(Deform(p))).} \tag{2.23}$$

Tapering Deformations

A tapering deformation gradually thins or expands an object along a chosen dimension. A tapered superquadric is shown in Fig. 2.15(a).

(a) (b)

Figure 2.15: Global deformations. (a) Tapering deformation and (b) bending deformation.

The tapering deformation is defined as

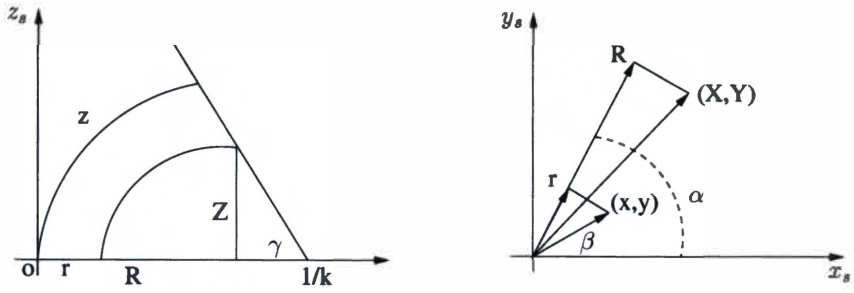
$$\begin{aligned} X &= f_x x = (\frac{k_x}{a_3} z + 1)x, \\ Y &= f_y y = (\frac{k_y}{a_3} z + 1)y, \\ Z &= z, \end{aligned} \tag{2.24}$$

where Variables k_x and k_y represent tapering factors along the x and y axes, respectively, in the object-centered coordinate system. with $-1 \leq k_x, k_y \leq 1$. Coordinates (x, y, z) define a point on the surface of a regular superquadric, and (X, Y, Z) define the corresponding point on the tapered superquadric surface.

Bending Deformations

The bending deformation defined by (Solina and Bajcsy, 1990) was specified for superquadric representation, i.e., recovering bent superquadric models from 2D images of objects. Differing from the bending deformation defined by Barr (Barr, 1984), which only permits bending in a single plane, Solina and Bajcsy introduced an additional parameter α to the bending deformation, which allows for bending in an arbitrary plane that goes through the z axis of the object coordinate system. A bent superquadric is shown in Fig. 2.15(b). The bending deformation is defined as

$$\begin{aligned} X &= x + (R - r)\cos\alpha, \\ Y &= y + (R - r)\sin\alpha, \\ Z &= (\frac{1}{k} - r)\sin\gamma, \end{aligned} \tag{2.25}$$



(a) (b)

Figure 2.16: Plots of bending deformations. Modified from (Solina and Bajcsy, 1990). (a) Bending deformation of the z axis and (b) bending deformation of a surface point.

where k represents curvature of the bending plane, and α the bending angle, both of which are specified by the user. Other parameters, γ, r, and R, are evaluated as

$$\begin{aligned} \gamma &= z \times k, \\ r &= \sqrt{x^2 + y^2}\cos\theta, \\ R &= \frac{1}{k} - (\frac{1}{k} - r)\cos\gamma, \end{aligned} \tag{2.26}$$

where

$$\theta = \alpha - \beta \qquad \text{and} \qquad \beta = \arctan\frac{y}{x}, \tag{2.27}$$

and variable θ is the angle between the vector $\vec{r}(x, y)$ and the bending plane.

The bending deformation is performed by projecting the surface points (x, y) of the undeformed model onto a bending plane, which is determined by the bending angle α. Next, the superquadric spine along the original z axis is bent to be a circular section. The bending degree is determined by curvature k. Both variables α and k are specified by users. Finally, the points on the bending plane are projected back onto the original plane to form the new (X, Y). Fig. 2.16(a) shows the bending deformation of the z axis. The length of the superquadric spine remains the same as the size of the undeformed model in the z axis (a_3). Fig. 2.16(b) shows the bending deformation of a surface point. The projection r can then be calculated by Equation (2.26). After bending each surface point onto the bending plane, the bending deformation is performed by evaluating R from Equation (2.26). Finally, the deformed point on the bending plane is projected back onto the original plane. The deformed point (X, Y, Z) on the original plane is evaluated by Equation (2.25).

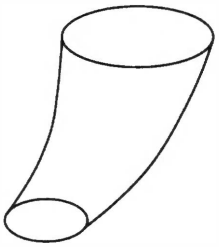


Figure 2.17: A tapered and bent superquadric.

Combination of Tapering and Bending Deformations

Since matrix multiplication is in general not commutative, the composition of deformations is not commutative (Jaklič et al., 2000), that is

$$\text{Bend(Taper(p))} \neq \text{Taper(Bend(p))}. \tag{2.28}$$

Equation (2.28) has shown that deformations acting on prototypical shapes have a specific order such that the transformation is more structure-preserving. The following structure is used for performing both tapering and bending deformations by

$$\text{Translation(Rotation(Bend(Taper(p))))}. \tag{2.29}$$

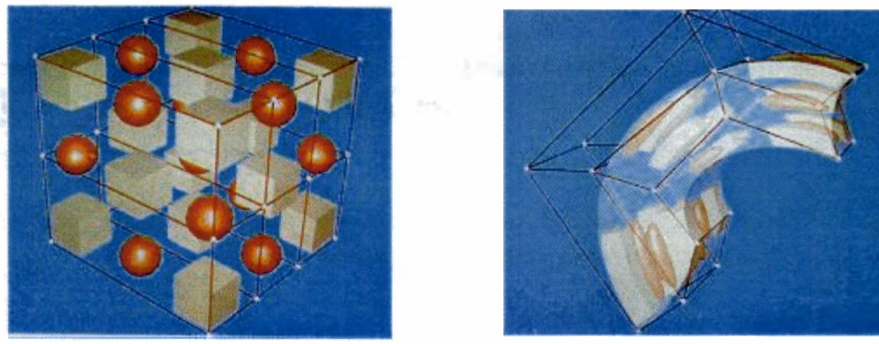
The inside-outside function for superquadrics in general position with tapering and bending deformations has a total of 15 parameters (Jaklič et al., 2000) expressed as

$$\Lambda_2 = (a_1, a_2, a_3, \varepsilon_1, \varepsilon_2, \phi, \theta, \psi, p_x, p_y, p_z; k_x, k_y, k, \alpha), \tag{2.30}$$

where the first 11 parameters define a regular superquadric. Parameters k_x and k_y define a tapering deformation, and parameters k and α define a bending deformation. By performing the tapering deformation defined in Equation (2.24), and bending deformation defined in Equation (2.25), a superquadric is globally deformed as shown in Fig. 2.17.

2.5.3 Free-Form Deformations (FFDs)

The free-form deformation (FFD) was first introduced in computer graphics (Sederberg and Parry, 1986), and later applied to represent and track medical data with superquadrics (Bardinet et al., 1998). FFD is a type of deformation lying between global and local deformations. The global shape of a solid with FFDs is determined by an initial superquadric model while simultaneously the local details of the surface can be described by the FFDs by adjusting a set of user-specified control points iteratively. The main benefit of FFDs is that the resulting deformation of the object is only defined by a small number of points, instead of the displacement of every model point defined in typical methods involving local deformation (Terzopoulos and Metaxas, 1991). Therefore, FFDs can represent local geometry of surfaces precisely with



(a) (b)

Figure 2.18: Free-form deformations. The two images are obtained from (Sederberg and Parry, 1986). (a) Undeformed object elements and defined control points inside the parallelpipedic box and (b) deformed control points and objects.

low representation costs. Another important characteristic of FFDs is that a parametric surface remains parametric after deformation, which is suitable for superquadric representation.

The basic principle of FFDs is summarized as follows. The object to be deformed is embedded in a 3D box. Inside this box, a volumetric grid of points are defined as control points to link the box to the object by a pre-defined polynomial, i.e., a deformation function as shown in Fig. 2.18 (a). The box is then deformed with the displacement of the control points, and the deformation is transmitted to the object as shown in Fig. 2.18(b). The FFD formulation is divided into two steps: (i) computation of local coordinates of the object points in the coordinate system as defined by the box of composed control points, and (ii) displacement of the control points and estimation of the new positions of the deformed object points.

The original deformation function (Sederberg and Parry, 1986) was defined as a tensor product of trivariant Bernstein polynomials. The position X of an arbitrary object point is evaluated by its local coordinates (s, t, u), the deformation function, and the current position of the control points P_{ijk} as

$$X = \sum_{i=0}^{L}\sum_{j=0}^{M}\sum_{k=0}^{N} B_{i,L}(s)B_{j,M}(t)B_{k,N}(u)P_{ijk}, \tag{2.31}$$

where $B_{i,L}(s)$, $B_{j,M}(t)$ and $B_{k,N}(u)$ are the Bernstein polynomials. The Bernstein polynomial is defined as

$$B_{i,L}(s) = C_L^i s^i (1-s)^{L-i} = \frac{L!}{i!(L-i)!} s^i (1-s)^{L-i}. \tag{2.32}$$

2.5.4 Objective Functions in Superquadric Fitting

Superquadric recovery is essentially a data fitting process. The objective function provides an error metric between the raw data and the recovered model. Given an error metric as the

objective function, a standard non-linear least square optimization approach, the *Levenberg-Marquardt* routine, is mostly used to minimize the fitting error (objective function) between the raw data and the recovered models. Four objective functions which have been primarily used in the literature are introduced as follows.

Objective Function Based on Definition. The objective function (Solina and Bajcsy, 1990) based on the inside-outside function of superquadrics is expressed as

$$G_1(\Lambda) = a_1 a_2 a_3 \sum_{i=1}^{N} (F^{\epsilon_1}(x_c, y_c, z_c) - 1)^2, \tag{2.33}$$

where x_c, y_c and z_c are coordinates of a point in the canonical system. The item $a_1 a_2 a_3$ is positively proportional to the volume. This results in the superquadrics having the minimum volume. From Equation (2.17), the following can be derived

$$F^{\epsilon_1}(x_c, y_c, z_c) - 1 = \frac{d}{|r_s|}\left(\frac{d}{|r_s|} + 2\right), \tag{2.34}$$

where r_s is the point on the superquadric surface as well as on the line through point P and the center of the superquadric. Therefore, the objective function expressed in Equation (2.33) can be changed to

$$G_1(\Lambda) = (a_1 a_2 a_3) \sum_{i=1}^{N} \left(\frac{d_i}{|r_s|}\left(\frac{d_i}{|r_s|} + 2\right)\right)^2. \tag{2.35}$$

This equation implies that data points inside the superquadric (i.e. $d_i < 0$) contribute less to the fitting error than those outside the superquadric with the same $|d_i|$ and $|r_s|$. Furthermore, this causes bias fitting to the data sets containing outliers (Dop, 1999). The volume factor $a_1 a_2 a_3$ only partially compensated this bias according to (Gross and Boult, 1988).

Objective Function Based on Radial Euclidean Distance. The objective function (Dop, 1999) based on radial Euclidean distance is expressed as

$$G_2(\Lambda) = \sum_{i=1}^{N} d^2 = \sum_{i=1}^{N} (|r_0||1 - F^{-\frac{\epsilon_1}{2}}(x_c, y_c, z_c)|)^2, \tag{2.36}$$

where d is the radial Euclidean distance between a point and a superquadric model. The literature stated that in some cases, this function recovered more accurate models than when using $G_1(\Lambda)$, but it sometimes completely failed to capture dimensions of the objects.

Objective Function with Background Constraints. This objective function is formed by adding background constraints to $G_2(\Lambda)$ (Dop, 1999). In doing so, the new equation becomes

$$G_3(\Lambda) = \sum_{i=1}^{N}(d_i^2 + w_k m(d_i')^2), \text{where } \begin{cases} m(d_i') = d_i', & \text{when } d_i' < 0 \\ m(d_i') = 0, & \text{otherwise} \end{cases}, \tag{2.37}$$

where w_k is the weight of the penalty at the kth iteration. This penalty is increased at every iteration to ensure the final recovery includes minimal constraining points, i.e. does not intersect the estimated supporting plane. Variable d_i' is the estimated supporting plane. During model recovery, a penalty to a point is attached to the estimated supporting plane d_i' when the point falls inside the superquadric at the previous iteration. When the constraining point is outside the surface, no penalty is assigned to d_i'. This algorithm results in dramatic improvement of superquadric recovery (Dop, 1999). However, robustness still needs to be added to handle the outlier problem.

Robust Objective Function. Robustness can be added to $G_3(\Lambda)$. The final objective function (Dop, 1999) becomes

$$G_4(\Lambda) = \sum_{i=1}^{(100-x)N}((d_i^2)_{i:N} + w_k m(d_i')^2), \text{where } \begin{cases} m(d_i') = d_i' & \text{when } d_i' < 0 \\ m(d_i') = 0 & \text{otherwise} \end{cases}, \tag{2.38}$$

where $(d_i^2)_{1:N} \leq \ldots \leq (d_i^2)_{N:N}$. The parameter x is the expected maximum percentage of outliers in the data set. The recovery performance of this error metric is improved significantly. The drawback is that it is difficult to determine the weight function.

Comparing these four objective functions, the first two objective functions based on the inside-outside function and the radial Euclidean distance, are selected in this dissertation due to their direct implementations and stable performances.

2.6 Summary

This chapter has presented a comprehensive review of the research literature with the emphasis on superquadric representation. The theoretical background on regular superquadrics, global deformations, and FFDs were introduced as well. A variety of objective functions used in the literature to recover superquadrics were introduced, and their performances were discussed. We conclude that polygonal meshes are capable of representing arbitrary shapes and local details on object surfaces. However, polygonal meshes are scale-dependent, and higher-level surface characterization must be explicitly maintained with meshes (Campbell and Flynn, 2001). In addition, data reduction and compactness achieved in polygonal models are insufficient for higher-level tasks such as object recognition (Bajcsy et al., 1990).

As a volumetric primitive, superquadrics have a lot of advantages. Compared with generalized cylinders which are mainly recovered from intensity images, superquadrics can be successfully recovered from range images or associated surface meshes, which can reflect accurate

geometric characteristics of objects. In addition, compared with a large number of vertices and triangles contained in a surface mesh for an object, superquadrics uniquely define the object with only 15 parameters. This significantly lower storage cost of superquadrics enables computer vision tasks such as visualization, data transmission, manipulation, etc. to be performed in real time. Moreover, the part-based parametric nature of superquadrics make them especially suitable for object recognition. Although an individual superquadric cannot represent arbitrary shapes, multiple superquadrics combined in a combinatorial manner can represent complex shapes such as a distributor cap. Global deformations can also improve the representation ability of superquadrics. The drawbacks of existing superquadric representation methods are summarized as follows.

- Existing methods have recovered superquadrics from single-view images containing partial surfaces of objects, which causes low confidence of the recovered superquadrics.
- Existing methods cannot represent complex, multi-part objects or multi-object scenes with superquadrics efficiently and reliably.

To address these two problems, a multi-view superquadric representation approach is proposed in this dissertation to improve the confidence and accuracy of superquadrics recovered from single-view range images. In addition, a new part decomposition algorithm is developed within the multi-view superquadric representation approach to facilitate superquadric representation of multi-part objects and multi-object scenes.

Chapter 3

Multi-View Superquadric Representation Algorithm

The primary contribution of this dissertation is the development of the multi-view superquadric representation algorithm for multi-part objects and multi-object scenes. Two problems existing in the research literature motivate this research. First, superquadric representation of multi-part objects or multi-object scenes has been an unsolved problem due to the complex geometry of objects. Second, superquadrics recovered from single-view range data tend to have low confidence and accuracy due to partially scanned object surfaces caused by inherent occlusions. To address these two problems, this dissertation presents a multi-view superquadric representation algorithm. By incorporating both part decomposition and multi-view range data, the proposed algorithm is able to not only represent multi-part objects or multi-object scenes, but also achieve high confidence and accuracy of recovered superquadrics. The abilities of representing complex objects and scenes while achieving high confidence of recovered superquadrics are major advantages of the proposed multi-view superquadric representation algorithm.

In this chapter, we present the multi-view superquadric representation algorithm. We first introduce multi-view strategies that have appeared in computer vision in Section 3.1 as a motivation for the proposed multi-view representation approach. Next, the multi-view superquadric representation algorithm is presented in 3.2. Next, each step of the multi-view superquadric representation algorithm including part decomposition, superquadric fitting, view registration, and view integration is described in Sections 3.3 through 3.6. Finally, we conclude this chapter by summarizing both advantages and limitations of the proposed multi-view superquadric representation algorithm in Section 3.7.

3.1 Occlusions and Multi-View Strategies

The interest in reconstruction of the surface geometry and topology of physical objects for use in reverse engineering and computer graphics has grown in recent years. Most 3D surface reconstruction methods use range data as input due to its high accuracy compared with stereo data. Multi-view scans as shown in Fig. 3.1(a) are essential for surface reconstruction of an office scene as shown in Fig. 3.1(b) due to occlusions and incomplete information inherent

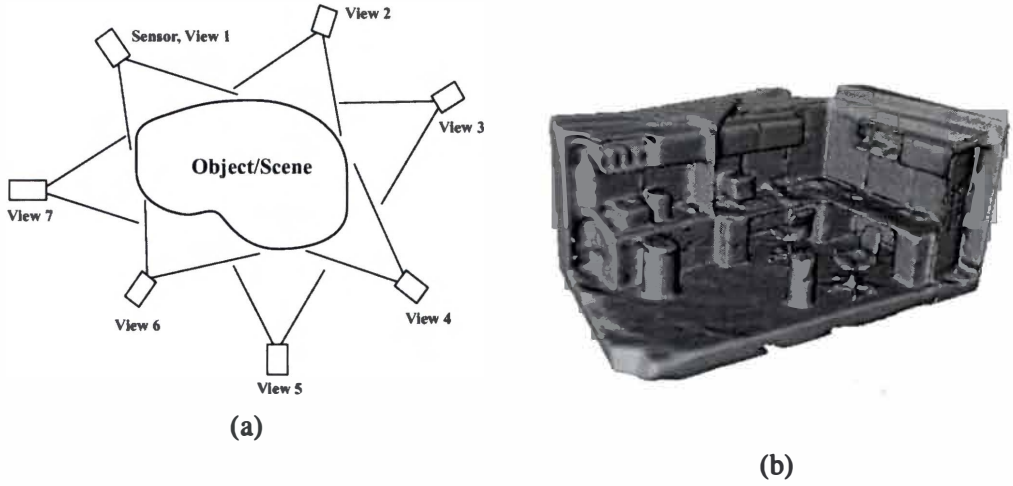
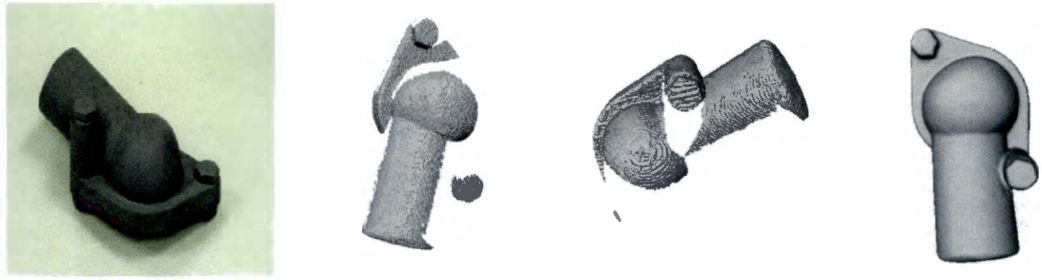


Figure 3.1: An office scene represented by triangle meshes. The mesh is reconstructed from multiple scans. (a) Sensor placement around objects/scenes and (b) rendering of original mesh.

in a single-view range image. For range images, occlusions arise either when the reflected laser light does not reach the camera, or when the directed laser light does not reach the scene surface (Maver and Bajcsy, 1993). Either no data or outliers are obtained for occluded areas after a scan is taken. Occlusions are classified into object occlusions where parts of the objects are hidden in viewpoints due to object self-occlusions, and shadow occlusions where a surface region is visible from one viewpoint but not in another (Polleyfeys and Gool, 1999). We classify occlusions which occur in a single-view range image into three categories according to the complexity of objects or scenes of interest as (i) self-occlusions of a single-part object, (ii) occlusions between different parts of a multi-part object, and (iii) occlusions between different objects in a multi-object scene.

Self-occlusions occur when only part of a single-part object is visible and scanned from a single view. Other invisible parts of the object from this view are occluded by the object itself. Fig. 2.11 displays a simple, single-part cylindrical object on the floor scanned from two different viewpoints. Fig. 2.11(a) shows that only one side of the object is scanned from the first view, and the rest of the object is occluded from this view. In Fig. 2.11(b), the other side of the object is scanned from the second view, and the rest of the object cannot be scanned and is invisible from this view. For complicated, multi-part objects, not only self-occlusions occur for each single part, but occlusions exist between different parts. Fig. 3.2 exhibits two views for a multi-part object, the automotive water neck. Fig. 3.2(b) shows that from the first view, many occluded surfaces between the ball and the base and between the small screw and the handle appear as dark (no data) areas. From Fig. 3.2(c), we observe that a big part of the handle of the water neck is occluded by the small cylinder next to it, and the ball is partially occluded since no laser light can reach this area due to the viewpoint chosen.

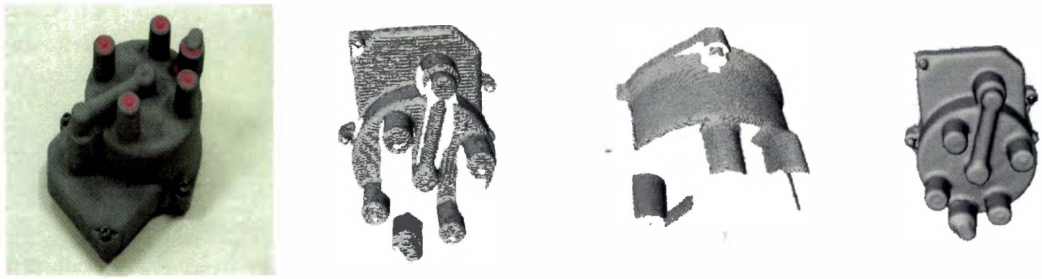


(a) (b) (c) (d)

Figure 3.2: Occlusions occurred in range scans for a water neck. The range images are acquired and calibrated into meshes from the IVP Ranger System. Calibrated data are represented by triangle meshes. The 3D model is reconstructed from 28 views, and consists of 58,784 vertices and 117,564 triangles. (a) Photograph of original object, (b) rendering of original mesh from view 1, (c) rendering of original mesh from view 2, and (d) rendering of reconstructed mesh.

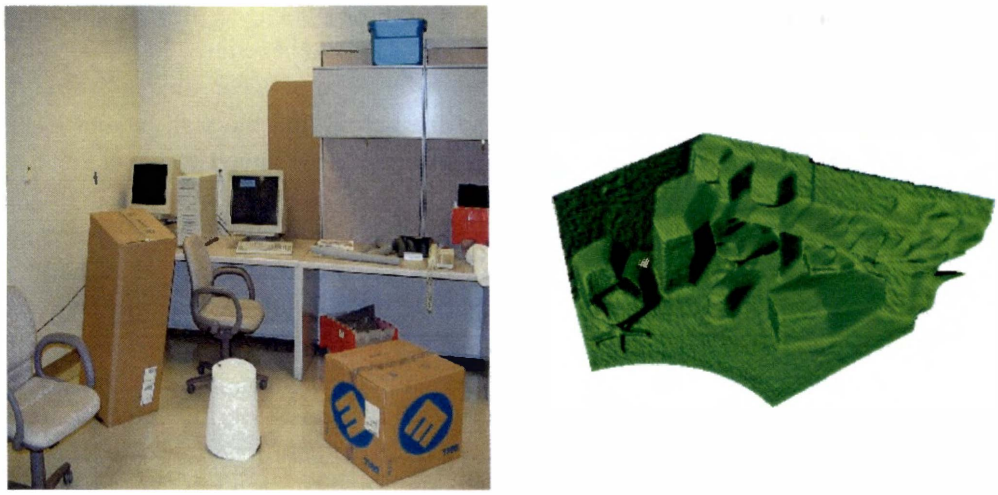
Fig. 3.3 displays a multi-part object, distributor cap scanned from two viewpoints. Fig. 3.3(b) shows the scanned data from view 1. We can observe that most of the upper surface of the base is occluded by the small cylinders on top of it. These areas appear as no data in the mesh. Also, each small cylinder at the top of the cap contains self-occlusions. Fig. 3.3(c) illustrates that most surfaces of the distributor cap are occluded, and are invisible from view 2. This figure demonstrates that most parts of an object could be occluded when scanning local details of the object. Occlusions in multi-object scenes are similar to those contained in multi-part objects. Some objects might be visible from one view but not from another view. Fig. 3.4 shows an office scene scanned from a single view containing occlusions between objects. This figure illustrates that the floor behind the shorter box and the barrel, and the wall behind the taller box are completely occluded. These occluded areas appear as no-scanned points in the mesh as shown in Fig. 3.4(b).

Multi-view strategies are used extensively to solve the occlusion problems in 3D computer vision tasks involving range images, e.g., surface reconstruction from multiple range images. As shown in Fig. 3.5, different parts of objects that are only visible to a single view can be registered and integrated into more complete data. In this manner, occlusions and ambiguities for the objects contained in a single-view scan can be reduced. A typical surface reconstruction task using multiple range images includes four stages (Hilton et al., 1996): (i) multi-view range image acquisition, (ii) multi-view data registration, (iii) data integration or 3D model building, and (iv) model optimization. The corresponding diagram is displayed in Fig. 3.6. The first step, the multi-view range data acquisition, deals with viewpoints selection, i.e., utilizing the least number of accessible viewpoints to reconstruct watertight models. This viewpoint selection problem is also defined as the Next Best View (NBV) problem (Pito, 1999). To be specific, the NBV problem is to determine which areas of a scanner's viewing volume need to be scanned to sample all of the visible surfaces of an a priori unknown object, and where to position/control the scanner to sample them (Pito, 1999). Pito presented a method determining the unscanned



(a) (b) (c) (d)

Figure 3.3: Occlusions occurred in range scans for a distributor cap. The range images are acquired and calibrated into meshes from the IVP Ranger System. Calibrated data are represented by triangle meshes. The 3D model is reconstructed from 15 views, and consists of 58,975 vertices and 117,036 triangles. (a) Photograph of original object, (b) rendering of original mesh from view 1, (c) rendering of original mesh from view 2, and (d) rendering of reconstructed mesh.



(a) (b)

Figure 3.4: Occlusions occurred in range scans for an office scene. The range images are acquired and calibrated into meshes from the RIEGL scanner. Calibrated data are represented by triangle meshes. (a) Photograph of original scene and (b) rendering of original mesh.

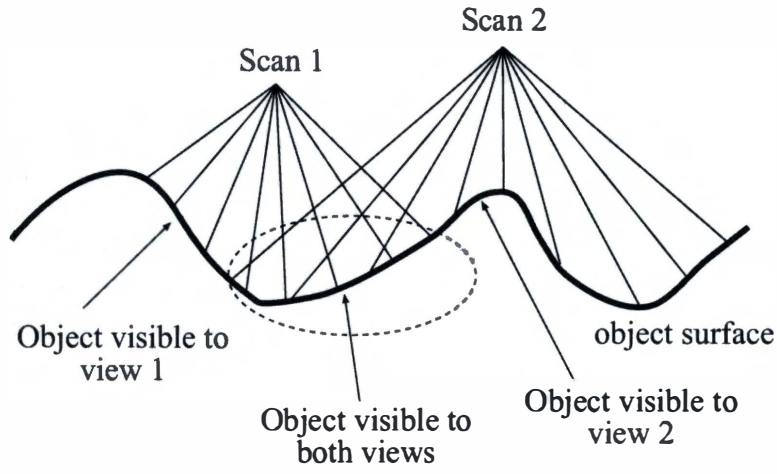


Figure 3.5: Scans from two views. Overlapping object surfaces scanned from two views need to be registered in order to be integrated into more complete data.

Figure 3.6: The pipeline of a 3D surface reconstruction system.

areas of the viewing volume. Other research addressing the NBV problem includes (Connolly, 1999; Maver and Bajcsy, 1993; Tarabanis et al., 1995; Whaite and Ferrie, 1997a; Yuan, 1995).

Since each of the four stages in a surface reconstruction task is both labor-intensive and time-consuming, most researchers in this area concentrate on only one of them. The NBV problem, by itself, is arduous to implement. Therefore, the surface reconstruction task, in reality, chooses sensor placement in a different way other than the numerical NBV solution. In most cases, viewpoints are first arranged heuristically around the object based on experience. The initial sensor placement significantly depends on complexity of the object and the working environment. For instance, some viewpoints may be inaccessible. During the data acquisition process, redundant viewpoints are usually employed to reconstruct watertight models, i.e., smooth surfaces without any missing patch. The surface reconstruction process is a back-and-forth process with frequent interactions between data capturers and model builders. For example, the mesh for the distributor cap shown in Fig. 3.3(d) is reconstructed from 15 views. The reconstructed mesh contains 117,036 triangles. The mesh for the water neck shown in Fig. 3.2(d) is reconstructed from 28 views. The 3D reconstructed mesh contains 117,564 triangles.

Inspired by the multi-view strategies for 3D surface reconstruction, we apply a similar technique to superquadric representation. Consequently, we propose a multi-view superquadric representation algorithm in Section 3.2. However, our algorithm differs from surface reconstruction methods in several aspects. First, the ultimate goals of the two types of algorithm are different. For a surface reconstruction task, the ultimate goal is to create complete surfaces of real objects. Therefore, multiple overlapping views are usually needed to reconstruct surfaces with the details of the objects. In contrast, our multi-view superquadric representation algorithm aims to represent objects with superquadrics in high confidence and accuracy. Therefore, overlaps between views are not necessary due to the volumetric nature of superquadrics. Second, due to the fact that superquadrics can be recovered from partial surfaces, our algorithms needs much less views. For instance, 28 views are needed to reconstruct the 3D mesh for the water neck as shown in Fig. 3.2(d) while only six views might be sufficient to recover superquadrics of high confidence and accuracy for the water neck.

3.2 Overview of the Algorithm

According to the literature review in Section 2.2, existing superquadric representation methods have been utilizing single-view images, which cannot guarantee the high confidence and accuracy of recovered superquadrics. Since recovered superquadrics are subject to the viewpoint from which the scan is taken, the final decision on the recovered superquadrics cannot be made only based on various single-view scans. Moreover, superquadric representation of multi-part objects or multi-object scenes has been an unsolved problem due to the complex geometry of objects. To address these problems, we propose a multi-view superquadric representation algorithm. By incorporating both part decomposition and multi-view range data, the proposed algorithm is able to not only represent multi-part objects or multi-object scenes, but achieve high confidence and accuracy of recovered superquadrics.

The multi-view superquadric representation algorithm consists of (i) part decomposition and initial superquadric model recovery from single-view range data, (ii) pairwise view registration based on recovered superquadric models, (iii) view integration, (iv) part decomposition of integrated data, and (v) final superquadric fitting for each decomposed part. The diagram of the algorithm is illustrated in Fig. 3.7. The major advantages of our proposed multi-view superquadric representation algorithm are summarized as follows.

- Both regular and deformable superquadrics can be handled.
- Superquadrics with high confidence and accuracy are obtained.
- Multi-part objects and multi-object scenes can be successfully represented due to the utilization of part decomposition.

The selection of optimal viewpoints belongs to the next best view (NBV) problem (Pito, 1999) in the path-planning category, and is beyond the scope of this research. The number of views needed in our multi-view superquadric representation approach depends on the complexity of objects or scenes to be represented. Due to the volumetric nature of superquadrics, several views are usually sufficient to recover correct superquadrics. Given a captured image from the first view, to obtain the maximum non-overlapping information, the second view should be opposite to the first view as shown in Fig. 3.8. Our multi-view representation algorithm first uses these two views to recover superquadrics. More views need to be incorporated if the recovered superquadrics from the first two views are either incomplete, i.e., the number of recovered superquadrics is unequal to the number of objects, or are inaccurate compared with the ground truths of the objects. The procedures contained in the multi-view superquadric representation algorithm are introduced in the following sections.

3.3 Part Decomposition

Superquadric representation of multi-part objects or multi-object scenes has been an unsolved problem due to the complex geometry of objects. Fig. 2.5 illustrates that an individual superquadric can only represent a single-part solid. Therefore, in order to represent multi-part objects with superquadrics, the objects need to be first decomposed into single parts. Although the superquadric representation method proposed in (Leonardis et al., 1997) can directly recover superquadrics for multi-part objects from range images without pre-segmentation, we observe that this method is inadequate and insufficient for our applications in which more complex objects and scenes are of interest. Consequently, we incorporate part decomposition into our multi-view superquadric representation algorithm as a pre-segmentation to first decompose multi-part objects into single parts. Superquadrics can then be recovered for each decomposed part. The diagram of superquadric representation utilizing part decomposition is illustrated in Fig. 3.9. It can be observed that by incorporating part decomposition as a pre-segmentation step, the original superquadric representation of complex objects is simplified into a superquadric fitting problem.

Since superquadrics are volumetric primitives, decomposed parts as opposed to segmented regions can truly reflect the part-based volumetric nature of superquadrics. For this reason,

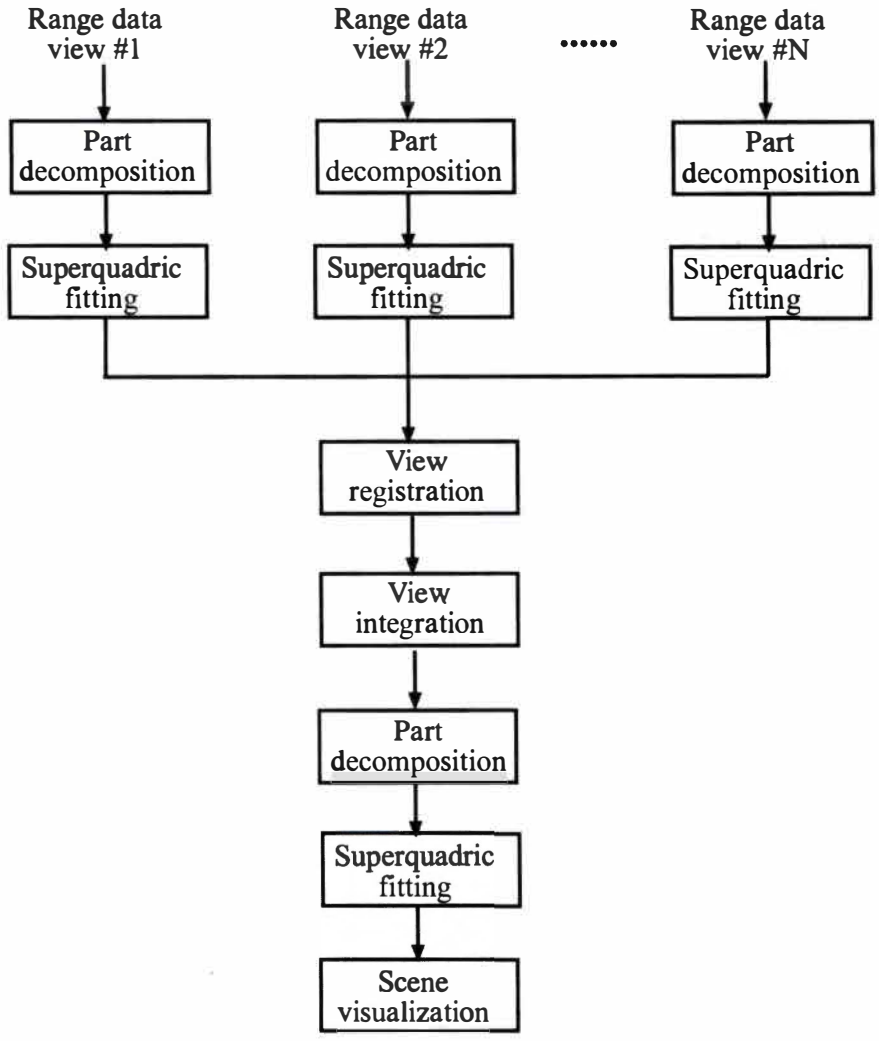


Figure 3.7: Diagram of the multi-view superquadric representation algorithm. The 3D triangle meshes from range images are used as input. Part decomposition is utilized for multi-part objects or multi-object scene. The number of views depends on the complexity of the objects or scenes and the tasks involved. View integration is performed by RapidForm software.

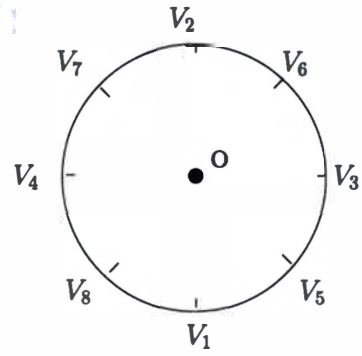


Figure 3.8: Viewpoints to scan objects. V_1 represents the first view, V_2 represents the next best view, etc.

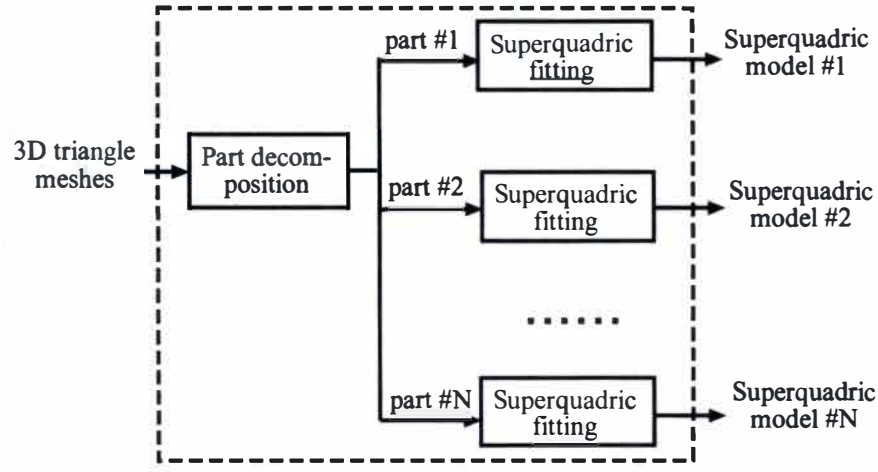


Figure 3.9: Diagram of superquadric representation for multi-part objects. The approach utilizes part decomposition.

(a) (b)

Figure 3.10: Part decomposition results for a teapot. This mesh is a reconstruction from Hughes Hoppe at Microsoft Research. The mesh consists of 3,042 vertices and 6,026 triangles while the decomposition consists of five parts. (a) Rendering of original mesh and (b) decomposed parts labeled in different colors.

we choose part decomposition instead of region segmentation as a presegmentation step in our multi-view superquadric representation algorithm. Part decomposition classifies data points on object surfaces into groups, each of which corresponds to one physical part of the object. The parts are consistent with human cognition. An example of part decomposition of a teapot using our part decomposition algorithm is shown in Fig. 3.10. The decomposition results demonstrate that our part decomposition algorithm decomposes a teapot into five parts including the cap, the handle, the body, the spout, and the small block at the bottom, which is consistent with human perception.

We propose a 3D part decomposition algorithm based on the transversality regularity theory (Koenderink and Doorn, 1982; Hoffman and Richards, 1984). The proposed part decomposition algorithm segments triangle meshes of objects into single parts. It can handle both single-view and reconstructed meshes. The diagram of the proposed part decomposition algorithm is shown in Fig. 3.11. Two assumptions are made for the proposed part decomposition algorithm.

- The input data is in the format of triangle meshes. The triangulated surfaces do not have to be watertight, i.e., holes or partial surfaces are acceptable.
- The input mesh is reasonably smooth. The smoothness of the mesh plays an important role in curvature estimation and boundary detection within the multi-view superquadric representation algorithm.

Starting from a 3D surface represented by smooth triangle meshes, Gaussian curvature is first estimated for each vertex on the surface. Neighborhood information contained in triangle meshes is utilized in this procedure. Next, based on the transversality regularity shown in Fig. 2.10, a user-defined threshold is applied to label vertices with highly negative curvature as boundaries or conjunctions between articulated parts. The remaining vertices are labeled as

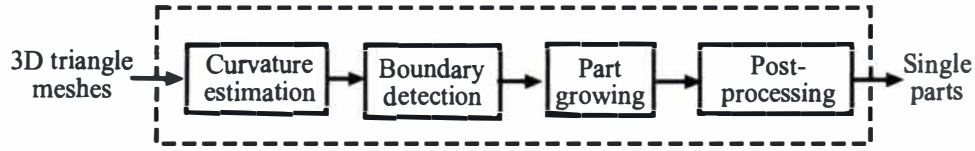


Figure 3.11: Diagram of the proposed part decomposition algorithm.

seeds belonging to potential object parts. After that, a pre-processing is performed to eliminate isolated vertices. Two types of isolated vertices defined in this work include (i) a point which is labeled as boundary while its neighbors are labeled as seeds and (ii) a point which is labeled as a seed while all of its neighbors are labeled as boundary. The labels of isolated vertices are changed to the same as their neighbors. A part growing operation is then performed on each seed vertex. In this procedure, each seed vertex is iteratively grown by including its neighbors which have the same label into a part until the part is surrounded by boundary vertices. Various parts are labeled from one to the number of parts contained in the object. Vertices within the same part have the same labels. Finally, a post-processing is applied to assign non-labeled vertices to one of the parts, and eliminate parts composed of too few vertices.

Strengths of the proposed part decomposition are summarized as follows.

- It can be easily implemented. The major steps in this algorithm include curvature estimation and part growing.
- It is efficient and reliable. It takes the algorithm about five minutes to decompose a reconstructed mesh for a distributor cap consisting of 58,975 vertices and 117,036 triangles. The algorithm demonstrates stable performances for the objects in the experiments.

More detailed description of the proposed part decomposition algorithm is provided in Chapter 4.

3.4 Superquadric Fitting of Single-Part Objects

For the multi-view superquadric representation algorithm shown in Fig. 3.7, after parts are decomposed from original objects, the 3D superquadric fitting is applied to each part decomposed from either single-view or integrated data. The recovery of superquadric models is essentially a data fitting process. Given an error metric as the objective function, the Levenberg-Marquardt routine (Press et al., 1992), which is a standard non-linear least square optimization approach, is used in this dissertation to minimize the fitting error between data and models.

The implicit definition for superquadrics in Equation (2.9) is defined in an object-centered coordinate system. A homogeneous transformation is applied to transform the model into the world coordinate system. The transform matrix consists of six parameters: $\phi, \theta, \psi, p_x, p_y$, and p_z. The first three define the rotation, and the others define the translation. Therefore, a globally

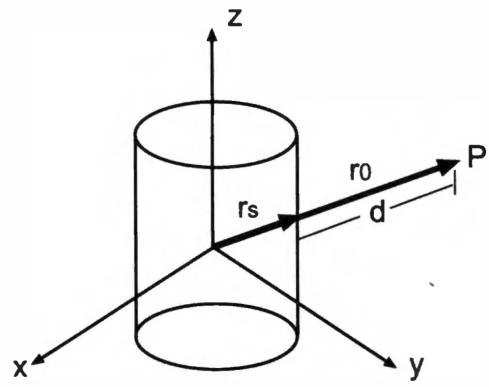


Figure 3.12: Radial Euclidean distance for superquadrics. The distance is between point P and a superquadric. d is the distance on the line through P and the center of the superquadric.

deformed superquadric model can be represented using a set of 15 parameters as

$$\Lambda = \{a_1, a_2, a_3, \varepsilon_1, \varepsilon_2, \phi, \theta, \psi, p_x, p_y, p_z, k_x, k_y, k, \alpha\}, \tag{3.1}$$

where the first 11 parameters define a regular superquadric, k_x and k_y represent tapering parameters, and k and α represent bending parameters. We have implemented superquadric fitting using Levenberg-Marquardt optimization technique (Press et al., 1992). An objective function based on the radial Euclidean distance (Gross and Boult, 1988) is used to evaluate how well the recovered model (superquadric) fits the raw data. This objective function is given as

$$G(\Lambda) = \sum_{i=1}^{N} d^2 = \sum_{i=1}^{N} (|r_0||1 - F^{-\frac{\varepsilon_1}{2}}(x_c, y_c, z_c)|)^2, \tag{3.2}$$

where d represents the radial Euclidean distance between a point and the corresponding superquadric model, and is depicted in Fig. 3.12. Variable F is defined in the implicit definition of superquadrics in Equation (2.9), and (x_c, y_c, z_c) represents coordinates expressed in the canonical coordinate system for a surface point of the superquadric.

For the global deformations described in Section 2.5.2, tapering deformations can be implemented in a straightforward manner. However, we observe that the original definition of bending deformations (Solina and Bajcsy, 1990) for superquadric representation is only valid for 2D image coordinate systems, in which the left bottom corner of a 2D image is the origin, and all the image pixels have coordinates of both $x > 0$ and $y > 0$. With the original definition of bending deformations, correct bent superquadrics cannot be rendered in an arbitrary 2D or 3D space, nor can they be recovered from 3D unstructured data points using the original definition, in both cases, x and y can be arbitrary values. To solve this problem, we propose a *quadrant analysis* technique to render and recover bent superquadrics in arbitrary 3D space.

From the bending plane and original points in four quadrants shown in Fig. 3.13, we derive correct equations to calculate θ.

Original point (x, y) in the first quadrant. For undeformed point (x, y) in the first quadrant, as shown in Fig. 3.13(a), the point (x, y) could be under or above the bending plane as (x_1, y_1) and (x_2, y_2) respectively. Variable θ, the angle between the vector (x, y) and the bending angle, is calculated as

$$\theta = \begin{cases} \beta_1 - \alpha & \text{for point } (x_1, y_1) \\ \alpha - \beta_2 & \text{for point } (x_2, y_2), \end{cases} \tag{3.3}$$

where $\beta_1 = \arctan \frac{y_1}{x_1}$, and $\beta_2 = \arctan \frac{y_2}{x_2}$. Therefore, $\cos\theta$ in Equation (2.26) can be evaluated by $\cos\theta = \cos(\beta - \alpha)$.

Original point (x, y) in the second quadrant. For the original point (x, y) in the second quadrant, as shown in Fig. 3.13(b), θ is calculated by

$$\theta = \begin{cases} \pi + \beta_1 - \alpha & \text{for point } (x_1, y_1) \\ \pi + \beta_2 - \alpha & \text{for point } (x_2, y_2), \end{cases} \tag{3.4}$$

where β_1, β_2 have the same meanings as in Equation (3.3). Therefore, $\cos\theta$ in Equation (2.26) can be evaluated by $\cos\theta = -\cos(\beta - \alpha)$.

Original point (x, y) in the third quadrant. For undeformed point (x, y) in the third quadrant, as shown in Fig. 3.13(c), θ is calculated by

$$\theta = \begin{cases} \pi + \beta_1 - \alpha & \text{for point } (x_1, y_1) \\ \pi + \beta_2 - \alpha & \text{for point } (x_2, y_2), \end{cases} \tag{3.5}$$

where β_1, β_2 have the same meanings as in Equation (3.3). Therefore, $\cos\theta$ in Equation (2.26) can be evaluated by $\cos\theta = \cos(\pi + \beta - \alpha) = -\cos(\beta - \alpha)$.

Original point (x, y) in the fourth quadrant. For undeformed point (x, y, z) in the fourth quadrant, as shown in Fig. 3.13(d), θ is calculated by

$$\theta = \begin{cases} \alpha - \beta_1 & \text{for point } (x_1, y_1) \\ \alpha - \beta_2 & \text{for point } (x_2, y_2), \end{cases} \tag{3.6}$$

where β_1, β_2 have the same meanings as in Equation (3.3). Therefore, $\cos(\theta)$ in Equation (2.26) can be evaluated by $\cos\theta = \cos(\beta - \alpha)$.

In summary, θ can be calculated by the following equations according to which quadrant the undeformed surface point lies in.

$$\theta = \begin{cases} \alpha - \beta & \text{for points in quadrants I and IV,} \\ \pi + \beta - \alpha & \text{for points in quadrants II and III.} \end{cases} \tag{3.7}$$

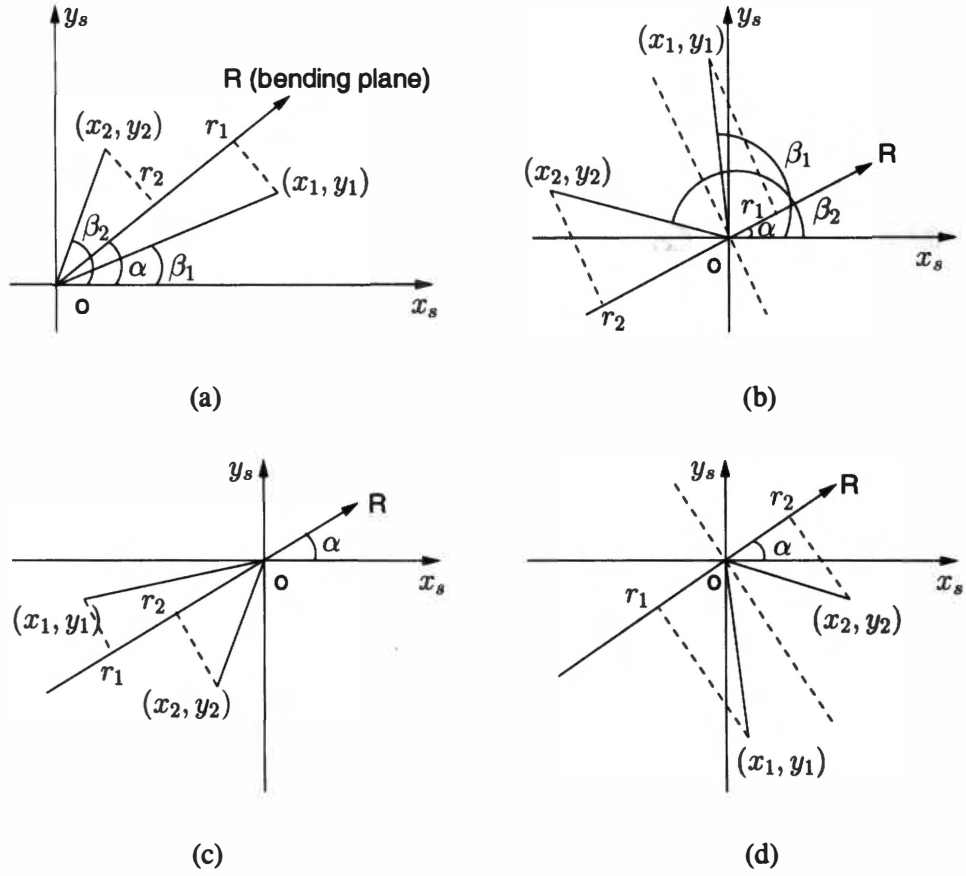


Figure 3.13: Projection of a point onto a bending plane. Coordinates (x_1, y_1) and (x_2, y_2) represent original points. Variables r_1 and r_2 are corresponding projections on the bending plane. The point is in (a) quadrant I, (b) quadrant II, (c) quadrant III, and (d) quadrant IV.

Parameters, γ, r, and R, are then evaluated as

$$\begin{aligned} \gamma &= z \times k, \\ r &= \sqrt{x^2 + y^2}\cos\theta, \\ R &= \frac{1}{k} - (\frac{1}{k} - r)\cos\gamma, \end{aligned} \tag{3.8}$$

and the coordinates for a point on the surface of a bent superquadric can then be evaluated as

$$\begin{aligned} X &= x + (R - r)\cos\alpha, \\ Y &= y + (R - r)\sin\alpha, \\ Z &= (\frac{1}{k} - r)\sin\gamma, \end{aligned} \tag{3.9}$$

where k represents curvature of the bending plane, and α specifies the bending angle, both of which are specified by the user.

We have implemented the algorithm for recovering single superquadrics with free-form deformations (FFDs) (Bardinet et al., 1998) as a subsequent step following the part decomposition procedure in our superquadric representation approach. The algorithm starts from a set of unstructured 3D points obtained from object surfaces. An initial superquadric coarsely capturing the global shape of the object is first recovered. Next, a free-form deformation is performed on the initial superquadric to fit the data precisely by adjusting the control points iteratively. The deformation terminates when the difference between two successive superquadric models is less than a pre-defined threshold. The algorithm is summarized as follows.

Algorithm 1 (Recovery of single superquadrics with free-form deformations)

Input: A set of 3D unstructured points on the surface of a single object.

Step 1. Recover an initial coarse superquadric model.

Step 2. Define a parallelpipedic box embedding the initial superquadric model, and choose the corresponding local coordinate system. Control points are calculated as lattices of the box. The local coordinate system is shown in Fig. 3.14.

Step 3. Compute local coordinates of each superquadric point in the parallelpipedic box coordinate system, and evaluate deformation matrix based on Equation (2.31).

Iterate the following steps

Step 4. Compute displacement field δX_n between the superquadric and the object data. Then calculate $X_n^a = X_n + \delta X_n$.

Step 5. Compute new control points according to the displacement of the superquadric points by minimizing $\| BP_{n+1} - X_n^a \|^2$.

Step 6. Compute new deformed superquadric model by evaluating $X_{n+1} = BP_{n+1}$.

Stop criterion: Least square error of the successive superquadrics less than a pre-defined threshold, i.e., $\frac{\|X_{n+1} - X_n\|}{\|X_n\|} <$ threshold.

Output: 11 parameters of the initial SQ and a set of updated control points.

Three coordinate systems are involved in this algorithm. In step 2, the local parallelpipedic coordinate system is defined as such: the origin is chosen as $(a_1, -a_2, a_3)$; (a_1, a_2, a_3) represents the sizes of the initial superquadric in three dimensions. The superquadric coordinate

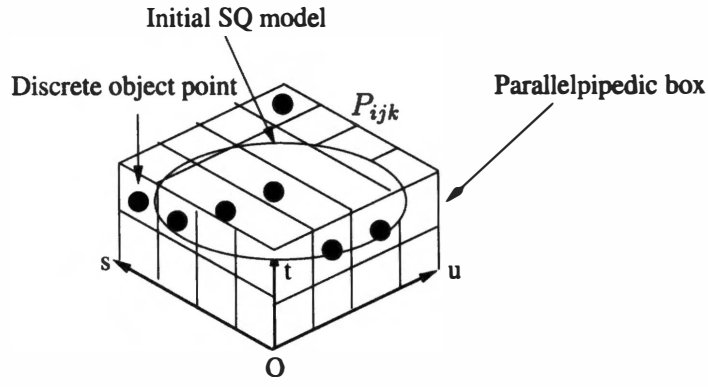


Figure 3.14: Local coordinates defined in the parallelpipedic box. The parallelpipedic box embeds the objects and the initial superquadric model. Vertex P_{ijk} represents a control point. Vector (s, t, u) represent the three axes of the local coordinate system.

system is defined as such: the origin is the center of the superquadric, and the (x, y, z) axis are the principle axes of the superquadric. The object points are defined in the world/object coordinate system. In step 3, (s, t, u) in Equation (2.31) represents coordinates of the model point X in the parallelpipedic coordinate system, and remains unchanged throughout the deformation. Points X and P_{ijk} are coordinates of the superquadric points and the control points in the superquadric coordinate system. In step 4, the object points must first be transformed into the superquadric coordinate system before the displacement fields between the object points and the superquadric points are evaluated. The transformation matrix can be calculated from the parameters of the superquadric as provided in Equation (2.15). In step 5, the single value decomposition (SVD) method is used to solve the least square minimization problem.

The output of the algorithm includes 11 parameters of the initial SQ model and a set of updated control points. With the 11 parameters, the superquadric can be rendered; furthermore, the deformation matrix can be calculated. With the current deformed control points and the deformation function (a matrix), the superquadric with FFDs can be derived. In this manner, this algorithm keeps the compactness of superquadrics while improving their representation abilities with FFDs. A more complete description of the algorithm can be found in (Bardinet et al., 1998).

3.5 View Registration

As a crucial step in multi-view strategies, view registration is to align data sets obtained from different views of the same object or scene together by analyzing the overlapping data between the different views. A typical view registration approach consists of two stages: initial coarse estimation of the transformation between views and refinement. We focus on two-view registration for rigid objects. In such cases, a homogeneous transformation exists between the two

views. The initial estimate of the transformation is crucial as the first step during the registration process. The posterior iterative closest point (ICP) refinement algorithm requires the initial estimate to be within an accuracy range to converge. There are three primarily used techniques to estimate the initial transformation: (i) feature correspondence-based , (ii) sensor parameter-based, and (iii) human operator-based. The latter two techniques utilize either a precisely calibrated scanning device or a GUI software to obtain the coarse transformation between views. These types of method heavily rely on available sensor systems and involve inefficient and inaccurate human manipulations. In contrast, the registration methods based on feature correspondences automatically extract features invariant to views, establish correspondences, and evaluate the transformation based on correspondent features. This type of method gains a lot of attention since it is purely automatic and data-driven.

Most image registration techniques are based on original images, and require a certain amount of overlap to successfully establish correspondences. For this reason, the computational complexity for this type of method is usually very high due to the extensive searching and optimization involved. In this dissertation, we have extended the view registration algorithm based on superquadrics (Jaklič et al., 2000) in both technical and application aspects. From the technical point of view, we first derive the first- and second-order inertial moments for a tapered superquadric to enable the registration algorithm to handle tapered as well as undeformed superquadrics. Second, we extend the initial 2D range image registration algorithm to 3D view registration. From the application point of view, the proposed view registration algorithm can be applied to register 3D scenes containing complex backgrounds and multiple objects. The only assumption of the proposed surface registration algorithm is that at least one object which can be represented by superquadrics should be in the scene, and the object is visible from views to be registered. As long as this condition is satisfied, the proposed view registration approach can be applied to register arbitrary scenes. This condition can be easily satisfied for most real indoor and outdoor scenes. An example can be found in a hazardous environment clean-up application (National Energy Technology Laboratory, deactivition & decommissioning focus area, 2002).

The proposed view registration algorithm based on superquadric representation has two major advantages over other surface registration methods based on feature correspondences. The first advantage results from the nature of superquadrics. In other words, since superquadrics can be successfully recovered from single-view, incomplete data sets, this view registration technique requires neither a time consuming point correspondence search nor a large amount of overlap between two views while most existing registration methods require at least 20% overlap (Chua and Jarvis, 1996; Roth, 1999). The second advantage is that since superquadrics can be successfully recovered from unstructured points without face information, no triangulation step is needed. Therefore, the proposed algorithm is free from the mesh resolution inconsistency problem that needs to be handled by an extra mesh uniform step in other registration approaches (Chen and Medioni, 1992; Johnson and Hebert, 1998). However, triangle meshes are necessary for our part decomposition algorithm. Once single parts are decomposed from the original objects or scenes, only unstructured point clouds are used for view registration.

Given a pair of 3D data sets to be registered, superquadrics are first recovered from each view. A new coordinate frame is then constructed based on recovered superquadrics for each

view, which is rigidly attached to the object in each set of data. After the two frames are constructed, their relationships to the world coordinate system can be calculated. Consequently, the rigid transformation between the two frames is estimated, and the two views are registered. The major step-by-step procedures for the view registration algorithm are summarized as follows.

Algorithm 2 (3D View registration based on superquadrics)

Input: Two 3D unstructured data sets in the format of (x, y, z).

Step 1. Recover superquadrics from each data set to be registered.

Step 2. Compute the first- and second-order inertial moments of each recovered superquadric in its canonical coordinate system.

Step 3. Transform the moments into the world coordinate system.

Step 4. Sum up the inertial moments of all the superquadrics recovered from each view and construct a new canonical frame for each view.

Step 5. Evaluate the rigid transformation between the two views.

Output: A homogeneous matrix representing the rigid transformation between the two input data sets (views).

In summary, the proposed view registration approach extends and improves the 2D range image registration approach (Jaklič et al., 2000) in the following aspects.

- Our view registration approach can register both regular and tapered superquadrics while the original method can only handle regular superquadrics. We derive the first- and second- order inertial moments for tapered superquadrics which are indispensable to superquadrics-based view registration.
- We extend the original 2D range image registration approach to register 3D unstructured data sets.
- We explore more generic and complicated scenes consisting of complex backgrounds and multiple objects.

More detailed description of the proposed view registration algorithm is provided in Chapter 5.

3.6 View Integration

View integration is a process of integrating multiple registered range data sets into a single 3D surface model which has topology and geometry consistent with the measurements. The range data sets are acquired from different viewpoints to overcome occlusions inherent in a single 2.5D range image. Surface reconstruction of objects with arbitrary topology has usually been performed on triangular meshes because polygonal meshes provide a simple piecewise planar surface representation that gives a first-order approximation without constraints on topology (Hilton et al., 1996). In addition, a triangulated mesh can be easily constructed for a single range image. Existing integration algorithms aim to merge constrained triangulations of individual range images.

Hilton et al. (Hilton et al., 1996) provided a comprehensive performance evaluation of existing integration algorithms, and classified them into four categories: (i) implicit surfaces using

points and normals (Hoppe and Derose, 1992), (ii) canonic subsets of the Venn diagram (Soucy and Laurendeau, 1995), (iii) mesh zippering (Turk and Levoy, 1994), and (iv) mesh growing (Rutishauser et al., 1994). The four integration algorithms are completely different in constructing a single triangulated model resulting in different complexity, limitations and failure models. Hilton et al. (Hilton et al., 1996) evaluated the four integration algorithms in terms of complexity analysis and inherent limitations. It has been concluded that the four integration algorithms being compared have the same order of complexity, approximately $O(M^2N)$ integrating M images of N points. In addition, the four integration algorithms were observed to have different limitations in terms of sample resolution, reconstructing surface topology at boundaries, small holes, regions of high curvature, etc.

We used a commercial software package RapidForm (INUS Technology, Inc, 2001) to integrate the multiple registered data sets. The mesh zippering method (Turk and Levoy, 1994) was used to integrate multi-view registered data sets. As shown in Fig. 3.7, after registering and integrating the multi-view data, part decomposition is performed, and superquadrics are recovered from each decomposed part. The superquadrics recovered from the integrated data have higher confidence and accuracy due to the utilization of multi-view data.

3.7 Summary

This chapter proposed a multi-view superquadric representation algorithm. The multi-view representation algorithm is able to improve confidence and accuracy of recovered superquadrics due to the greater amount of information utilized than with single-view images. In addition, this algorithm can represent multi-part objects and multi-part scenes. Corresponding experiments for the proposed multi-view superquadric representation algorithm are presented in Chapter 7. The proposed multi-view superquadric representation algorithm has been published in (Zhang et al., 2002a). The advantages of the proposed multi-view superquadric representation algorithm are summarized as follows.

- It can handle both regular and deformable superquadrics.
- It recovers superquadrics with high confidence and accuracy.
- It is capable of representing multi-part objects or multi-object scenes due to the utilization of part decomposition.

This algorithm extends the state of the art for superquadric representation in the sense of representing multi-part objects while achieving high confidence and accuracy. How much the confidence and the accuracy of recovered superquadric parameters can be improved from the proposed multi-view representation algorithm will be demonstrated by the experiments in Chapter 7.

Chapter 4

Part Decomposition Algorithm Based on Curvature Analysis

Part decomposition segments an object into its constituent parts consistent with human perception. Part decomposition is a crucial step within our proposed multi-view superquadric representation algorithm since it directly facilitates the superquadric representation of multi-part objects. In addition, part decomposition significantly simplifies the original difficult superquadric representation task into a straightforward superquadric fitting. Many tasks in computer vision, computer graphics, and reverse engineering are directly performed on objects or corresponding models. These tasks become more difficult when the object is complicated, for example, when it contains multiple parts. In such cases, part decomposition can simplify the problem of multi-part, complex objects into several sub-problems, each dealing with a single-part of the original object.

In this chapter, we present a novel part decomposition algorithm and associated superquadric representation of multi-part objects utilizing part decomposition. We begin with introducing motivations for part decomposition in Section 4.1. Several part theories are introduced. Next, we present a novel part decomposition algorithm for 3D triangle meshes in Section 4.2. Superquadric representation of multi-part objects utilizing part decomposition is presented as well. Curvature estimation and boundary detection of the part decomposition algorithm are presented in Section 4.3. Morphological operators performed on triangle meshes are also described. Section 4.4 presents part growing and post-processing of the part decomposition algorithm. Section 4.5 concludes this chapter with a summary of the proposed part decomposition and superquadric representation algorithms.

4.1 Motivations and Part Decomposition Rules

The literature review on superquadric representation indicates that most existing superquadric representation methods using 3D data can only handle single-part objects. For the multi-part objects as shown in Fig. 4.1, there is no efficient superquadric representation methods in the literature. Therefore, we develop a superquadric representation algorithm for representing multi-part objects with superquadrics following a part decomposition step. In this manner, the original

(a) (b)

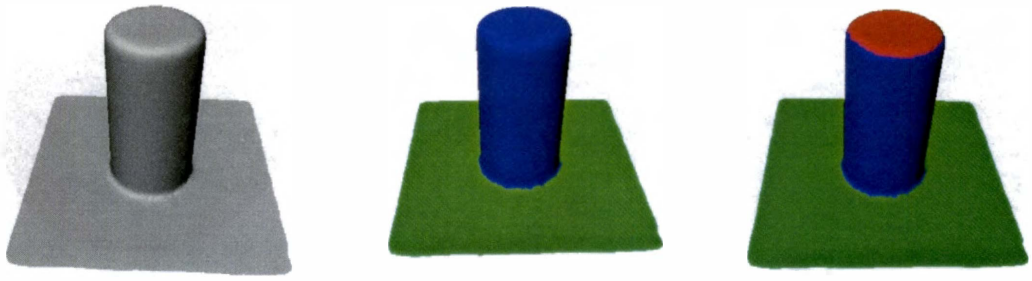
Figure 4.1: Multi-part objects. (a) Distributor cap and (b) water neck.

complicated representation problem is simplified. Based on the literature review on part decomposition in Section 2.3, we conclude that there is a lack of part decomposition algorithms which can handle triangle meshes as needed in our superquadric representation task. On the other hand, existing methods that can handle triangle meshes segment meshes into surfaces instead of meaningful parts. We therefore propose a novel part decomposition algorithm based on Gaussian curvature analysis.

Part decomposition classifies data points on object surfaces into groups, each of which corresponds to one physical part of the object. To make the goal of our proposed part decomposition algorithm clearer, Fig. 4.2 illustrates the difference between region segmentation and part decomposition. As illustrated in Fig. 4.2, a region segmentation algorithm segments the cylinder into three regions while the part decomposition segments the cylinder into one single part. We believe that part decomposition is more appropriate for high-level tasks such as object representation and recognition using volumetric primitives.

4.1.1 Transversality Regularity, Minima Rule, and Parabolic Curve

As suggested by Hoffman and Richards (Hoffman and Richards, 1984), part decomposition methods can be classified into primitive- or boundary-based. A primitive-based approach defines parts by their shapes, not by their contours of intersections. A boundary-based approach defines parts by their contours of intersection, not by their shapes. In particular, primitive-based approaches decompose objects into parts by measuring the shape similarity between image data and predefined part models. This type of approach can also be called object representation since the parts are represented by shape primitives when they are segmented. Compared to the primitive-based approaches, the boundary-based part segmentation approaches segment objects into parts without knowing in advance what the parts look like. This is more flexible because part segmentation is essentially a grouping process, for which high-level primitive representation is unnecessary. In summary, boundary-based part segmentation approaches have greater versatility than primitive-based methods (Hoffman and Richards, 1984).



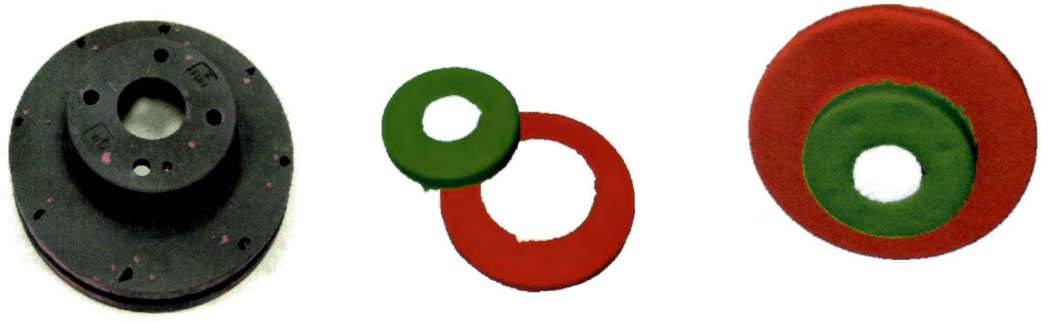
(a) (b) (c)

Figure 4.2: Difference between part and region segmentation for a scene. The scene consists of two linked objects: a cylinder and a plane. The mesh is created from two superquadrics, and consists of 27,381 vertices 54,420 triangles. Part decomposition results consist of two parts. Region segmentation results consist of three regions: two regions for the cylinder and one region for the plane. (a) Rendering of original mesh, (b) part decomposition results, and (c) region segmentation results.

The part theory used in boundary-based segmentation methods can be dated back to the *transversality regularity* proposed by Guillemin and Pollack (Guillemin and Pollack, 1974) as a human perception theory. When two arbitrarily shaped surfaces are made to interpenetrate they always meet in a contour of concave discontinuity of their tangent planes.

An example of transversality regularity is illustrated in Fig. 4.3. The disc brake is composed of two cylindrical parts penetrating each other as shown in Fig. 4.3. It can be observed that the intersection between the two parts forms a boundary composed of points with discontinuous, perpendicular tangents since these points belong to both parts. Two rules have been proposed for boundary-based part segmentation based on the theory of transversality regularity. Koenderink and Van Doorn (Koenderink and Doorn, 1982) presented parabolic curves as part boundaries. At a parabolic point (do Carmo, 1976), one of the principal curvatures of the surface changes from convex (> 0) to concave (< 0). The other rule is the *minima rule* proposed by Hoffman and Richards (Hoffman and Richards, 1984) to partition a surface into parts along all contours of concave discontinuity of the tangent plane. The minima rule (Hoffman and Richards, 1984) divides a surface into parts at loci of negative minima of each principal curvature along its associated family of lines of curvature.

Parabolic curves naturally lie on the boundaries between elliptic and hyperbolic regions (do Carmo, 1976). The only assumption is that the surface to be decomposed must be C^2 continuous, i.e., the curvature is defined everywhere. The parabolic curves could be either on the surface of the individual parts or on the border of the "glue" between parts. Based on the transversality regularity (Guillemin and Pollack, 1974), an *anticlastic* (negative Gaussian curvature) region almost always exists between convex parts when they are joined. Similar to (Rom and Medioni, 1994), which applied parabolic curves to part decomposition in intensity



(a) (b) (c)

Figure 4.3: An example of transversality regularity for a disc brake. Two parts (convex shapes) of the disc brake interpenetrate to create a contour of tangent discontinuity, i.e., a boundary between the two parts. (a) Photograph of the original object, (b) two separated parts, and (c) two parts penetrating each other to form a boundary.

images, we have applied the rule of parabolic curves (Koenderink and Doorn, 1982) to part decomposition for 3D triangle meshes due to its reliability. Another reason we choose parabolic curves is that Gaussian curvature can be easily estimated for triangle meshes while it takes much higher computation costs to estimate principle curvatures in the minima rule.

4.1.2 Surfaces and Curvatures

Surfaces can be distinguished according to their mean and Gaussian curvatures. In particular, surfaces are classified into elliptic (Gaussian curvature $K > 0$) and hyperbolic ($K < 0$). For elliptic surfaces, convex (mean curvature $H > 0$) and concave ($H < 0$) surfaces exist. To be more specific, surfaces can be classified into eight basic categories according to the mean and Gaussian curvature signs (Besl, 1988). The relationships between surface types and curvatures are tabulated in Table 4.1. Fig. 4.4 depicts the eight types of surfaces.

Gaussian curvature is an intrinsic property of a space independent of the coordinate system used to describe it. The Gaussian curvature of a regular surface in R^3 at a point p is formally defined as

$$K(p) = det(S(p)), \tag{4.1}$$

where S is the shape operator, and *det* denotes the determinant.

If $x : U \rightarrow R^3$ is a regular patch, then the Gaussian curvature is evaluated by (do Carmo, 1976)

$$K = \frac{eg - f^2}{EG - F^2}, \tag{4.2}$$

where E, F, and G are coefficients of the first fundamental form, and e, f, and g are coefficients of the second fundamental form.

Table 4.1: 3D surface types with various curvature signs. **K**: Gaussian curvature, **H**: mean curvature.

		K		
		-	0	+
H	-	Saddle ridge	Ridge	Peak
	0	Minimal surface	Flat	None
	+	Saddle valley	Valley	Pit

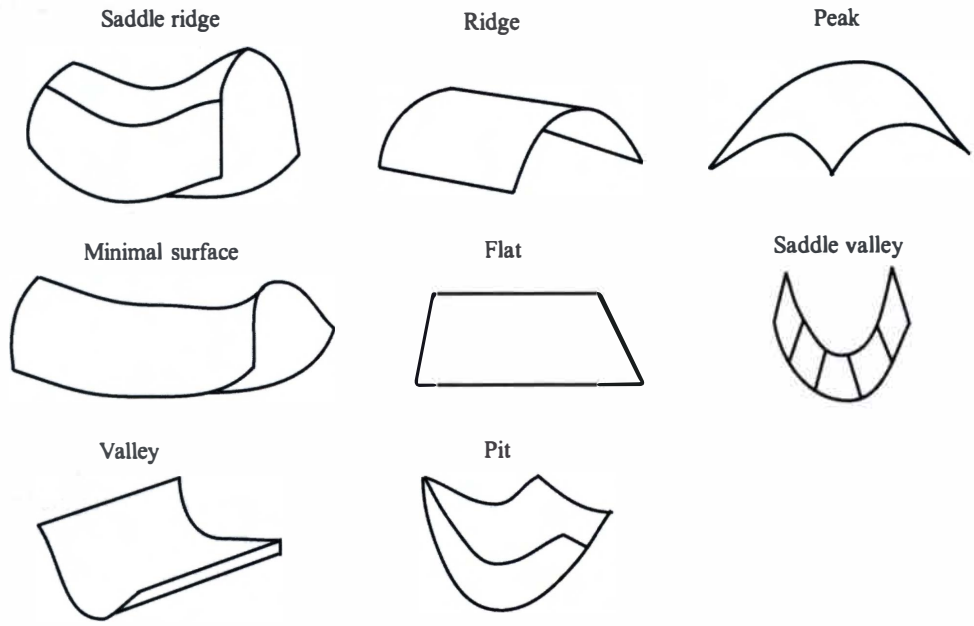


Figure 4.4: A variety of 3D surfaces with various curvature signs.

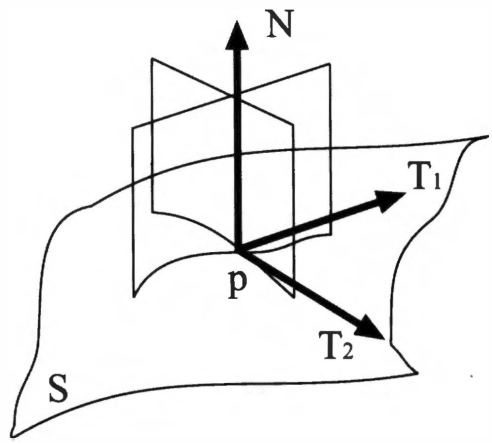


Figure 4.5: Principle directions on surfaces. Principle directions T_1 and T_2 at a point p on a surface s.

The Gaussian curvature can also be evaluated from principle curvatures as

$$K = k_1 k_2, \tag{4.3}$$

where k_1 and k_2 are principle curvatures measuring the maximum and minimum bending of a regular surface at each point. The associated principle directions at a point p on a surface S are depicted in Fig. 4.5.

The mean curvature is defined as

$$H = \frac{1}{2}(k_1 + k_2). \tag{4.4}$$

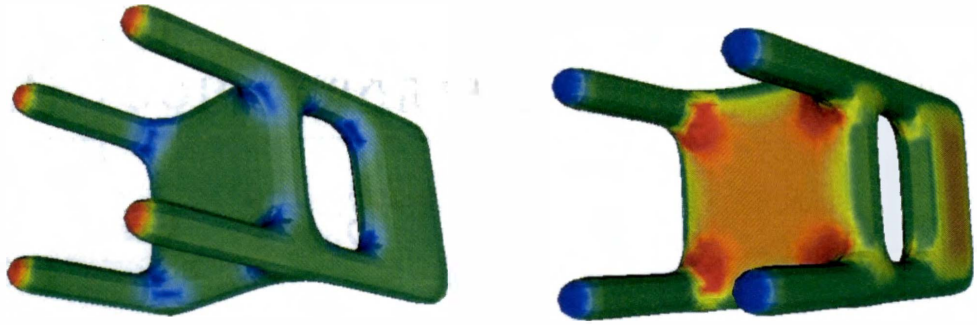
The Gaussian and mean curvatures satisfy

$$H^2 \geq K, \tag{4.5}$$

with equality only at umbilic points.

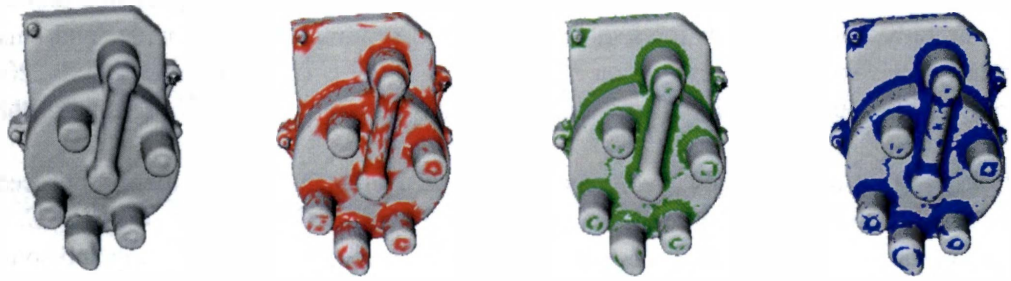
An example of a 3D model with labeled maximum and minimum Gaussian and mean curvatures is shown in Fig. 4.6. Fig. 4.6(a) illustrates that Gaussian curvature has negative values at the boundaries between two articulated parts, which proves the transversality regularity principal (Bennamoun, 1994). Mean curvature does not have such a characteristic for the model of the chair. However, from the surfaces shown in Fig. 4.4, we do observe that surfaces with negative mean curvatures could be boundaries between parts.

Fig. 4.7 shows a distributor cap and boundaries identified by both Gaussian and mean curvatures. This mesh is a reconstruction from multiple range images using the IVP Ranger System (Integrated Vision Products, 2000). The mesh consists of 58,975 vertices and 117,036 triangles. Fig. 4.7(c) shows that some part boundaries between the tube and the base of the distributor cap fail to be identified by Gaussian curvature. Fig. 4.7(d) shows that part boundaries between



(a) (b)

Figure 4.6: A curvature-labeled 3D model of a chair (Center for Vision, Speech, and Signal Processing, 2002). Red denotes maximum curvature and Blue denotes minimum curvature. (a) Gaussian curvature labeling and (b) mean curvature labeling.



(a) (b) (c) (d)

Figure 4.7: Identified boundaries for a distributor cap. This mesh is a reconstruction from multiple range images using the IVP Ranger System. The mesh consists of 58,975 vertices and 117,036 triangles. Both Gaussian and mean curvatures are applied to boundary identification. (a) Rendering of original mesh, (b) boundaries (labeled in blue) identified using Gaussian curvature, (c) boundaries (labeled in blue) identified using mean curvature, and (d) boundaries (labeled in blue) identified using both Gaussian and mean curvatures.

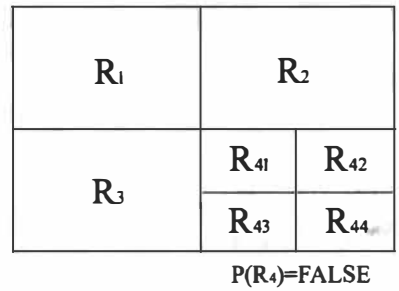


Figure 4.8: Quadrant region splitting. $P(R_4) = FALSE$.

one of the small cylinders and the base are missing using mean curvature. Fig. 4.7(e) shows part boundaries merged from Figs. 4.7(c) and 4.7(d). We observe that part boundaries identified using both Gaussian and mean curvatures yield complete part boundaries. Therefore, the proposed part decomposition algorithm utilizes both Gaussian and mean curvature to identify boundaries between linked parts.

4.1.3 Region Growing vs. Region Splitting and Merging

The objective of segmentation is to partition an image into regions. "Region growing" and "region splitting and merging" are two common methods to segment image pixels into different regions. The process of region splitting and merging initially subdivides an image into a set of arbitrary, disjointed regions. Then, the process merges and possibly splits these regions in an iterative manner to satisfy predefined constraints (Gonzalez and Woods, 2001). These constriants are developped next. Let R represent the entire image and P represent a predicate. One commonly used approach for segmenting R is to subdivide it into smaller and smaller quadrant regions so that, for any region R_i, $P(R_i) = TRUE$ (Gonzalez and Woods, 2001). Any region R_i is subdivided into its quadrant regions if $P(R_i) = FALSE$. This algorithm is illustrated in Fig. 4.8.

If only a splitting were used, the image tends to be over-segmented. A merging procedure is performed to merge two adjacent regions R_i and R_j only if $P(R_i \bigcup R_j) = TRUE$. The iterative region splitting and merging process terminates when no further merging and splitting is possible, i.e., $P(R_i) = TRUE$ for any segmented region R_i and $P(R_i \bigcup R_j) = FALSE$ for any two adjacent regions R_i and R_j.

Alternatively, region growing is a procedure that groups pixels or subregions into larger regions based on predefined criteria. The basic approach is to start with a set of "seed" points and from these seeds grow regions by appending neighboring pixels that have properties similar to the seeds (Gonzalez and Woods, 2001). The commonly used properties include gray level, gradient, color, etc. The selection of seed points usually depends on the specific problem. Every image pixel is taken as a seed point when no priori information about the image is available. The properties used to classify pixels depend not only on the problem under consideration, but also on the type of image data available. Color information can be used to group pixels in color

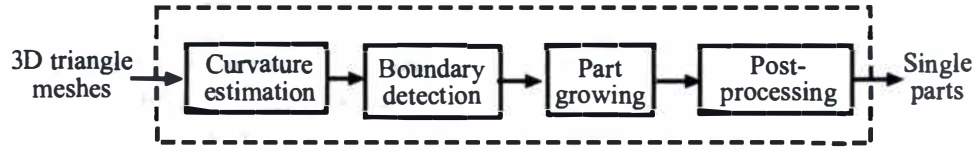


Figure 4.9: Diagram of the proposed part decomposition algorithm.

images while gray levels and spatial and geometric properties might be necessary when color images are unavailable. Generally speaking, growing of a region should stop when no more additional pixels satisfy the criteria for inclusion in that region (Gonzalez and Woods, 2001).

The above discussion on region growing and region splitting and merging are confined to 2D images. Our task is to segment a 3D triangle mesh into parts. The most important differences between 2D images and 3D meshes are bounds and neighbors. The bound (size) and neighborhood information are well-defined for a 2D image. In contrast, this information is difficult to obtain for a 3D mesh. A quadrant splitting process is easy and straightforward for a 2D image, but very time-consuming and inefficient for a 3D mesh. For instance, to perform a region splitting process on a 3D mesh, a 3D bounding box needs to be first constructed for the mesh, and the original model can be split into multiple parts evenly by comparing each vertex of the mesh with the bounding box. On the other hand, a region growing method can be extended to 3D mesh more easily due to the availablity of reliable third-party software programs developed for mesh manipulations. These software programs usually have well-constructed data structures and high performances. An example is the Computation Geometry Algorithms Libraries (CGAL) (Computational geometry algorithms library, 2002) developed by a group of European researchers. With these available softwares, fast neighborhood searching and other operations in a region-growing process for a 3D mesh can be performed easily. Therefore, we conclude that the region growing method is more feasible and efficient than region splitting and merging when applied to 3D triangulated models.

4.2 Overview of the Algorithm

The proposed part decomposition algorithm is a boundary-based algorithm where the transversality regularity principle and parabolic curves designate boundaries between articulated parts. The diagram of the proposed part decomposition algorithm is shown in Fig. 4.9. Two assumptions are made for the proposed part decomposition algorithm.

- The input data is in the format of triangle meshes. The triangulated surfaces do not have to be watertight, i.e., holes or partial surfaces are acceptable.
- The input mesh is reasonably smooth. The smoothness of the mesh plays important roles in curvature estimation and boundary detection within the multi-view superquadric representation algorithm.

Starting from a 3D surface represented by smooth triangle meshes, Gaussian curvature is first estimated for each vertex on the surface. Neighborhood information inherent in triangle meshes is utilized in this procedure. Next, based on the transversality regularity principle as shown in Fig. 4.3, a user-defined threshold is applied to label vertices with highly negative curvature as boundaries or conjunctions between two articulated parts. The remaining vertices are labeled as seeds belonging to potential object parts. After that, a preprocessing step is performed to eliminate isolated vertices. Two types of isolated vertices defined in this work include (i) a point which is labeled as a boundary while its neighbors are labeled as seeds, and (ii) a point which is labeled as a seed while its neighbors are labeled as boundary. The labels of isolated vertices are changed to the same labels as their neighbors.

A part growing process is then performed on each seed vertex. In this procedure, each seed vertex is iteratively grown by including its neighbors, which have the same label, into a part until the part is surrounded by boundary vertices. Various parts are labeled from one to the number of parts. Vertices within the same part have the same labels. Finally, a post-processing step is applied to assign non-labeled vertices to one of the parts, and to eliminate parts composed of too few vertices. This part decomposition algorithm is summarized as follows.

Algorithm 3 (Part decomposition of 3D triangle meshes)
Input: Triangulated 3D surfaces for an object or a scene.
Step 1. Compute Gaussian curvature for each vertex on the surface.
Step 2. Label vertices of highly negative curvature as boundaries using a pre-specified threshold. Label the remaining vertices as seeds.
Step 3. Eliminate isolated vertices.
Step 4. Perform iterative part growing on each seed vertex.
Step 5. Assign non-labeled vertices to parts, and eliminate parts having less than a pre-specified number of vertices.
Output: Decomposed parts consisting of point clouds written to separate files.

We incorporate part decomposition into our superquadric representation algorithm as a pre-segmentation to represent multi-part objects and multi-object scenes. A diagram of our superquadric representation algorithm utilizing part decomposition is illustrated in Fig. 4.10. We observe that by incorporating part decomposition as a pre-segmentation step, the original superquadric representation of complex objects is simplified into a superquadric fitting problem.

In addition, decomposed parts can truly reflect the part-based volumetric nature of superquadrics. For this reason, we choose part decomposition instead of region segmentation as a pre-segmentation step in our superquadric representation algorithm.

4.3 Curvature Estimation and Boundary Identification

Among the various curvature estimation methods reviewed in Section 2.3, we implement the mesh-based Gaussian curvature estimation method (Lin and Perry, 1982) due to its direct and fast implementation and reliable performance. Gaussian curvature for each vertex can be directly evaluated from the meshes with this method. Since only the sign of curvatures is used in

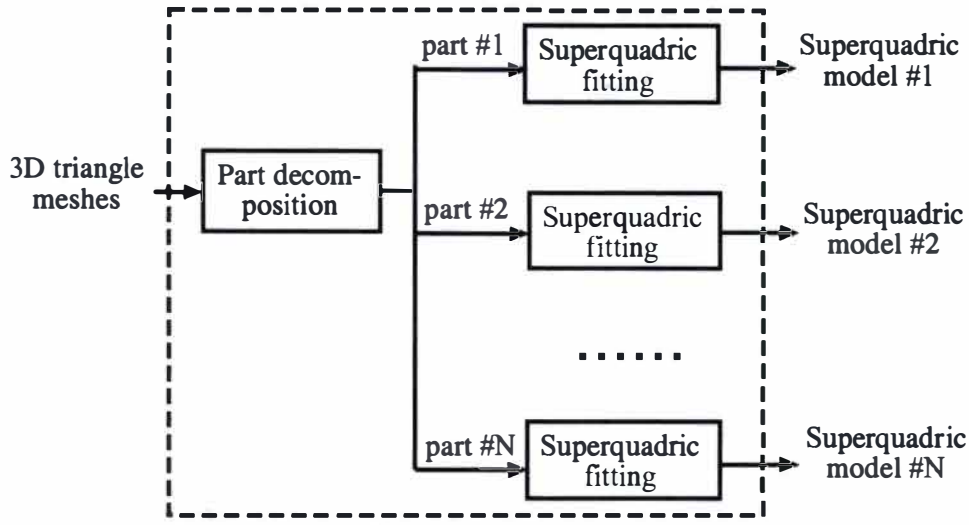


Figure 4.10: Diagram of superquadric representation for multi-part objects. Part decomposition is utilized.

our part decomposition algorithm, this discrete curvature estimation is sufficient for our application. As shown in Fig. 4.11, Gaussian curvature of the vertex p is computed as

$$K(p) = \frac{3(2\pi - \sum_{i=1}^{N} \theta_i)}{\sum_{i=1}^{N} A_i} \delta^2(p - p_i), \tag{4.6}$$

where p represents the point of interest, p_i represents one of the mesh neighbors of the point p, and A_i represents the area of the corresponding triangle. The variable θ_i represents the interior angle of the triangle at p and the variable $\delta()$ is the Dirac delta function.

By definition, the mean curvature H is the divergence of the surface around the normal vector, $H = \nabla \vec{n}$. For planar surfaces, the surface normals of the polygons around a vertex are the same, which results in zero curvature associated with that vertex. The mean curvature normal for a vertex p is defined as $H\vec{n} = \frac{\nabla A}{2A}$, where A is the area of a small region around the vertex $p(x, y, z)$ on the surface, and ∇ is the derivative with respect to the three coordinates. By this definition, the local area reaches minimum value on planar surfaces and results in zero mean curvatures for those vertices on the planar surfaces. For a surface mesh, a discrete formulation for the mean curvature can be derived from the continuous form. We estimate the mean curvature for a surface mesh according to the definition proposed by Desbrun et al. (Desbrun et al., 1999) as

$$-H\vec{n} = \frac{1}{4A} \sum_{j \in N(i)} (cot(\alpha_j) + cot(\beta_j))(\vec{x_j} - \vec{x_i}), \tag{4.7}$$

where j is the jth element in the vertex x_i's one-ring neighbor vertex set $N(i)$. The vector $(\vec{x_j} - \vec{x_i})$ indicates the edge e_{ij}, and α_j and β_j are the two inner angles in the two triangles

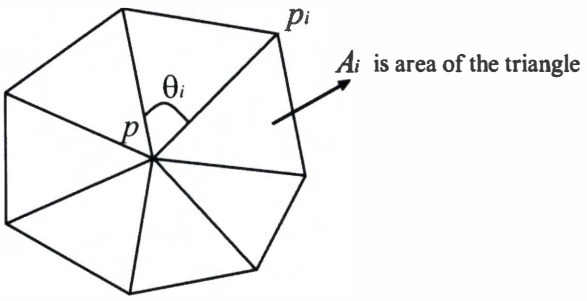


Figure 4.11: Gaussian curvature estimation for a vertex p. The vertex p_i is a neighbor vertex. The area of a neighbor triangle mesh is A_i.

sharing the edge e_{ij}. Variables α_j and β_j are also opposite to the edge e_{ij}. The variable A is the sum of the areas of the triangles sharing the vertex x_i. These variables are illustrated in Fig. 4.12.

After Gaussian and mean curvatures are obtained for each vertex on the surface, two specified thresholds are applied to label vertices as boundary (label 1) or seed (label 0). A vertex is identified as boundary if either its Gaussian or mean curvature is less than the associated threshold (highly negative values). Vertices of highly negative curvature are labeled as boundaries (0) while the rest are labeled as seeds (1) belonging to potential object parts. The labeling process is performed as the following.

$$L_{v_i} = \begin{cases} 0: & G_{v_i} < T_g \quad \text{or} \quad H_{v_i} < T_h \\ 1: & \text{else} \end{cases}, \tag{4.8}$$

where L_i denotes the label of vertex v_i, and G_{v_i} and H_{v_i} represent its Gaussian and mean curvature. The variables T_g and T_h are the Gaussian and mean curvature thresholds.

The thresholds are critical and affect the performance of subsequent part growing. The thresholds are subject to mesh resolution and smoothness, and are determined in a heuristic way. The Gaussian curvature threshold is determined as

$$T_g = \frac{c_g}{N} \sum_{v_i \in R_g} G_{v_i}, \tag{4.9}$$

where R_g is a union of vertices with negative Gaussian curvatures. The variable N is the number of vertices in R_g. Therefore, we choose c_g instead of the threshold T_g directly. We observe that $c_g \in [0.1, 0.3]$ yields good results for most application. Similarly, The mean curvature threshold can be determined as

$$T_h = \frac{c_h}{N} \sum_{v_i \in R_h} H_{v_i}, \tag{4.10}$$

where R_h is a union of vertices with negative mean curvatures. We observe that $c_h \in [0.2, 0.5]$ yields good results for most application.

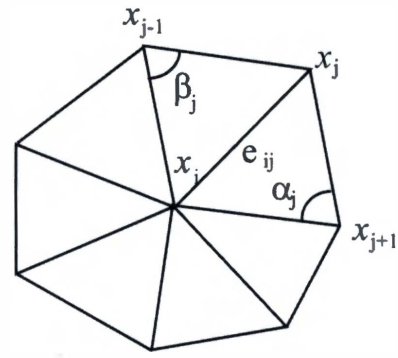


Figure 4.12: Mean curvature estimation for a vertex x_i. The neighbor triangles share the vertex x_i. Vertex x_j is one of the neighbor vertices for x_i. The variables α_j and β_j are two angles opposite to the edge e_{ij}.

According to the label of each vertex, isolated vertices are then removed. As noted previously, two types of isolated vertices defined in this work include (i) a point which is labeled as boundary while all of its neighbors are labeled as seeds and (ii) a point which is labeled as a seed while all of its neighbors are labeled as boundary. The labels of isolated vertices are changed to be the same as their neighbors.

The vertices labeled as boundary may not form a closed boundary, which causes problems for our part decomposition algorithm. In addition, the boundary identification process is sensitive to the resolution and smoothness of input meshes. To overcome the sensitivity, and create close boundaries, *mathematical morphology* is used to improve the labeling process. Mathematical morphology is reviewed briefly in 2D space and is extended to 3D meshes. In mathematical morphology, there are two fundamental operations of *dilation* and *erosion*, which lead to more powerful operations of *opening* and *closing*. The language of morphology is set theory. While A and B are sets in Z^2, the dilation of A by B is defined as (Gonzalez and Woods, 2001)

$$A \oplus B = \{z | [(\hat{B})_z \cap A] \subseteq A\}, \tag{4.11}$$

where set B is called the structuring element and the subscript z is displacement. The set $\hat{B}$ is the reflection of B and is defined as

$$\hat{B} = w | w = -b, \quad \text{for } b \in B. \tag{4.12}$$

Similarly, erosion is defined as

$$A \ominus B = \{z | (B_z) \subseteq A\}. \tag{4.13}$$

The opening of set A by structuring element B is defined as

$$A \circ B = (A \ominus B) \oplus B, \tag{4.14}$$

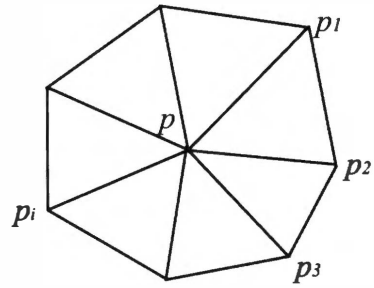


Figure 4.13: Neighborhood information contained in a triangle mesh. Point p_i represents a neighbor vertex of point p.

which denotes that the opening of A by B is the erosion of A by B, followed by a dilation of the result by B. Similarly, the closing of set A by structuring element B is defined as

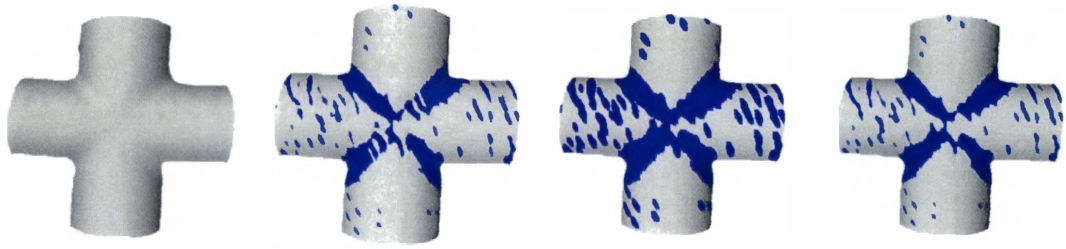
$$A \bullet B = (A \oplus B) \ominus B, \tag{4.15}$$

which says that the closing of A by B is the dilation of A by B, followed by a erosion of the result by B. Dilation expands an image and erosion shrinks it. Opening generally smoothes the contour of an object, breaks narrow isthmuses, and eliminates thin protrusions. Closing tends to smooth sections of contours but, as opposed to opening, it generally fuses narrow breaks and long thin gulfs, eliminates small holes, and fills gaps in the contour. More detailed descriptions of mathematical morphology applied to image processing can be found in (Gonzalez and Woods, 2001).

The definitions for the morphological operators defined in Equations (4.11)-(4.15) cannot be directly used on 3D triangle meshes since the topology of the mesh is unknown. Rössl et al. (Rössl et al., 2000) redefined binary morphological operators for triangle meshes, even though in a limited way.

The morphological operators defined in (Rössl et al., 2000) can only perform on binary values (0, 1) of vertices. In our part decomposition algorithm, the labeled values for each vertex after curvature thresholding are binary. The vertices are labeled either "1" (boundary vertex) or "0" (non-boundary vertex). The morphological operators are performed in the following way. The $d-$neighborhood is used as a structure element for every vertex. We are only concerned with one-ring neighborhood. Considering the triangle meshes in Fig. 4.13, if the label of vertex p is "1", the one-ring dilation is defined as changing the labels of all its neighbor vertices p_i to "1". On the other hand, the one-ring erosion is defined as changing the labels of neighbor vertex p_i to "0" if the label of vertex p is "0". As a result, the regions labeled as "1" are extended by dilations and are shrunk by erosions. Opening and closing are defined as combinations of dilation and erosion, the same as 2D space.

In addition, Rössl et al. (Rössl et al., 2000) demonstrated that dilation or erosion of an n-ring neighborhood equals to n times dilations or erosions of one-ring neighborhood according to their definitions. An example of dilation and erosion performed on triangle meshes is shown



(a) (b) (c) (d)

Figure 4.14: Morphological operators performed on a crossing-shaped pipe fitting. This mesh is a single-view scan from the IVP Ranger System. The mesh consists of 17,963 vertices and 35,238 triangles. Vertices with negative Gaussian curvatures are identified as boundary points and labeled blue. A closing operation is performed to link breaks and fill gaps in the boundary, which plays an important role in part decomposition. (a) Rendering of original mesh, (b) highly negative curvatures identified as boundaries and labeled blue, (c) boundaries after a one-ring dilation, and (d) boundaries after a one-ring erosion following the dilation.

in Fig. 4.14 for a crossing-shaped pipe fitting. Fig. 4.14(a) shows a rendered surface composed of triangle meshes. Fig. 4.14(b) illustrates that vertices with highly negative curvatures by a thresholding operation are identified as boundary points, and are labeled blue. Fig. 4.14(c) shows boundaries points after a one-ring dilation, and Fig. 4.14(d) shows eroded boundaries. Comparing the original boundary point and those after the closing operation (dilation followed by erosion), we observe that the closing operation can link breaks and fill gaps in the contour, which will benefit the succeeding steps in our part decomposition algorithm.

4.4 Part Growing and Post-processing

After the vertices are labeled, a part growing operation is performed on each vertex labeled as seed. Fig. 4.15 shows triangle meshes around point p. To illustrate the part growing process, a two-ring neighborhood of the mesh around point p is shown in this figure. Part growing is performed as follows. Starting from a seed vertex p, a unique part label is first assigned to the vertex. Second, the neighbors p_i initially labeled as seeds are then labeled with the same part number as the point p. The same labeling process is performed for each neighbor p_i to label vertices p_{ij}. This process terminates when the grown part is surrounded by boundary vertices, i.e., the neighbors of the edge vertices of the part are labeled as boundaries. This process is repeated for each seed vertex, but not for a vertex which has been grown and already labeled uniquely. After the seed vertices are assigned new labels, a refinement step is performed for each boundary vertex. Given a seed point x, its mesh neighbors x_i are first sorted in ascending order based on their Euclidean distances to the point x. Next, vertex x_k is selected from the ordered neighbors where x_k is the first vertex in the list with a part label and not a boundary label. The boundary vertex x is then labeled the same as the vertex x_k, i.e., the label of x is

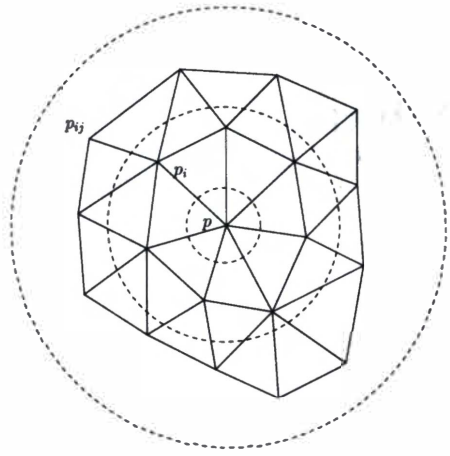


Figure 4.15: Two-ring triangle neighborhood around point p. The point p is labeled as a seed where point p_i represents a neighbor vertex of the point p. Point p_{ij} represents a neighbor vertex of the point p_i.

changed from boundary to the unique part label of x_k. Finally, with the exception of a few missing vertices, each vertex should now have a part label and thus be assigned to different object parts.

Finally, a post-processing step is performed to assign the remaining vertices that have not been labeled. For example, vertex p shown in Fig. 4.13 is an unlabeled vertex and needs further post-processing. Assuming that p_i $(i = 1, 2, ..., N)$ represents a neighbor vertex of the point p, the neighbor vertices are first selected if they have the same curvature sign as that of the vertex p and belong to one of the segmented parts. Next, among those neighbor vertices, the vertex which has the smallest Euclidean distance to the vertex p is selected as a target vertex. For example, vertex p_1 shown in Fig. 4.13 is assumed to be the target vertex of the vertex p. Finally, the vertex p is labeled the same as the vertex p_1, and the vertex p is further assigned to the same part as p_1. Furthermore, parts composed of fewer vertices than a specified threshold are merged with adjacent parts in a similar way.

4.5 Summary

We present a novel part decomposition algorithm on 3D triangle meshes. Two issues motivate this research. First, superquadric representation of multi-part objects aims to represent each single part of the original objects with a superquadric. These individual parts need to be decomposed from the original objects with a part decomposition algorithm. Second, there is no existing part decomposition algorithm for 3D triangle meshes in the literature. Therefore, we propose a boundary-based part decomposition algorithm utilizing curvature analysis. The primary characteristics of the algorithm are summarized as follows.

- The algorithm decomposes objects into individual parts instead of regions (surface patches). These parts are consistent with human perception. For instance, the algorithm decomposes a teapot into five parts which is consistent with human cognition according to the part theories.

- The algorithm uses triangle meshes as input. The mesh could be a single-view scan or reconstructed from multi-view range images. The triangulated surfaces do not have to be watertight, i.e., holes or partial surfaces are acceptable.

- The input mesh is assumed to be reasonably smooth. The smoothness of the mesh plays an important role in curvature estimation and boundary detection in the part decomposition algorithm.

- The algorithm contains two user-specified thresholds. One threshold identifies boundaries according to the Gaussian curvature. The other threshold specifies the minimum number of vertices contained in a segmented part. Both thresholds are determined heuristically.

With the aid of part decomposition, our superquadric representation algorithm can successfully represent multi-part objects or multi-object scenes. This algorithm extends the state of the art of superquadric representation in the sense of multi-part object representation and triangle mesh handling. The primary advantages and characteristics of the superquadric representation algorithm utilizing part decomposition are summarized as follows.

- The algorithm can represent multi-part objects or multi-object scene from 3D triangle meshes with the aid of an efficient part decomposition algorithm, which significantly extends the representation ability of superquadrics.

- The capability of handling 3D triangle meshes makes the algorithm flexible due to the popularity of triangle meshes in a large variety of research areas including reverse engineering, computer vision, computer graphics, etc. Superquadrics recovered from 3D data other than 2.5D range images reflect true geometry of the original objects. In addition, a set of 15 parameters defined for a superquadric can represent an object consisting of thousands of vertices and triangles in a mesh. The huge data reduction achieved by superquadrics enables many computer vision tasks such as visualization, data transmission, manipulation, etc. to be performed in real time. Moreover, the compact and parametric nature of superquadrics can significantly benefit object recognition.

- The algorithm incorporates global and free-form deformations, which significantly improves the representation ability and accuracy of superquadrics.

In this chapter, we presented a 3D part decomposition algorithm to partition an object into its constituent parts. We presented a superquadric representation approach of multi-part objects utilizing part decomposition. The part decomposition algorithm is boundary-based and utilizes the transversality regularity theory. The superquadric representation approach can represent multi-part objects or multi-object scenes. The proposed part decomposition algorithm has been

published in (Zhang et al., 2002b). Experimental results on part decomposition and succedent superquadric representation are shown in Section 7.1.

Chapter 5

View Registration Algorithm Based on Superquadrics

View registration is a crucial step within our proposed multi-view superquadric representation algorithm. In fact, view registration is indispensable to computer vision tasks involving multi-view images. For our multi-view superquadric representation algorithm, multi-view scans are acquired to reduce ambiguities or partial surfaces inherent in single-view range images. Each single-view scan is acquired in an independent local coordinate system. View registration aims to align the multi-view scans into a common global coordinate system by analyzing the overlapping data between the different views. We are only concerned with two-view (pairwise) registration of rigid objects for which a homogeneous transformation exists between the two views. A typical view registration approach consists of two stages: initial coarse estimation of the transformation between views and refinement of this transformation. The iterative closest point (ICP) algorithm (Besl and McKay, 1992; Zhang, 1994) is the most commonly used registration refinement algorithm. The initial estimate of the transformation needs to be within a constrained accuracy range for the ICP algorithm to converge. Three techniques are primarily used to estimate the initial transformation: (i) feature correspondence-based, (ii) sensor parameter-based, and (iii) human operator-based. The latter two techniques utilize either a precisely calibrated scanning device or a GUI software to obtain the coarse transformation between views. These types of method heavily rely on available sensor systems and involve inefficient and inaccurate human manipulations. In contrast, the registration methods based on feature correspondences automatically extract features invariant to views, establish correspondences, and evaluate the transformation based on correspondent features. This type of method gains a lot of attention since it is purely automatic and data-driven.

For our multi-view superquadric representation algorithm, superquadrics are first recovered for each single-view data. Therefore, we want to utilize the recovered superquadrics as much as possible in succeeding steps in the multi-view superquadric representation algorithm. For this reason, we improve and extend the range image registration algorithm (Jaklič, 1997), and present a 3D view registration algorithm based on superquadrics. In Section 5.1, we present the formulation of evaluating the first- and second- order inertial moments for tapered superquadrics. The new formulation is a crucial extension of our view registration algorithm.

In addition, inertial moments of superquadrics are indispensable to view registration based on superquadrics. We present our view registration algorithm in Section 5.2. Finally, Section 5.3 summarizes our view registration algorithm and concludes the chapter.

5.1 Inertial Moments of Tapered Superquadrics

Inertial moments of superquadrics are indispensable to view registration based on superquadrics. However, the inertial moments have only been derived for regular superquadrics in the literature. We formulate the evaluation of the inertial moments for tapered superquadrics as follows.

An object vector i and a matrix I of inertial moments are defined as

$$i = \begin{bmatrix} I_x \\ I_y \\ I_z \\ V \end{bmatrix} \quad \text{and} \quad I = \begin{bmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{xy} & I_{yy} & I_{yz} \\ I_{xz} & I_{yz} & I_{zz} \end{bmatrix}, \tag{5.1}$$

where I_x, I_y, and I_z represent the first-order inertial moments along the x, y, and z axes. The variable V represents the volume of the superquadric. The variable V represents the volume information. Each element in matrix I represents the second-order inertial moment. In a canonical coordinate system, the above vector and matrix change to

$$i_i^L = \begin{bmatrix} 0 \\ 0 \\ 0 \\ V_i \end{bmatrix} \quad \text{and} \quad I_i^L = \begin{bmatrix} I_{xx,i}^L & 0 & 0 \\ 0 & I_{yy,i}^L & 0 \\ 0 & 0 & I_{zz,i}^L \end{bmatrix}, \tag{5.2}$$

where the superscript L and the subscript i represent the canonical coordinate system and the i^{th} object respectively. As shown in Equation (5.2), the object vector i_i^L contains only volume information in the canonical system. The object matrix I_i^L contains the second-order inertial moments, $I_{xx,i}^L$, $I_{yy,i}^L$, and $I_{zz,i}^L$. The volume and the inertial moments of a regular superquadric can be calculated by Equations (2.19) and (2.21) as suggested by Jaklič (Jaklič, 1997).

We derive the volume and the inertial moments of a tapered superquadric as follows. The area of the cross section of a tapered superquadric, which is a superellipse, is computed as

$$\begin{aligned} A_t &= 2\int_0^{\pi/2} (X\dot{Y} - Y\dot{X}) d\omega \\ &= k_x k_y A = k_x k_y 2ab\varepsilon B(\frac{\varepsilon}{2}, \frac{\varepsilon+2}{2}), \end{aligned} \tag{5.3}$$

where A represents the area of the corresponding undeformed superellipse, and (X, Y) represents a point in the superellipse. Variables k_x and k_y represent tapering factors defined in Equation (2.24). Term $B(x, y)$ represents the Beta function defined in Equation (2.20). By converting the parameters a, b and ε of a superellipse into those of a superquadric, A_t can be

rewritten as

$$A_t = k_x k_y 2a(\eta)b(\eta)\varepsilon_2 B(\frac{\varepsilon_2}{2}, \frac{\varepsilon_2+2}{2}), \tag{5.4}$$

where

$$a(\eta) = a_1 \cos\eta^{\varepsilon_1} \qquad \text{and} \qquad b(\eta) = a_2 \cos\eta^{\varepsilon_1}. \tag{5.5}$$

By substituting the tapered factors k_x and k_y, and recalling the explicit definition of superquadrics defined in Equation (2.7), the corresponding differential volume can be expressed as

$$\begin{aligned} dV &= A(z)dz = A(\eta)\dot{z}(\eta)d\eta \\ &= 2a_1a_2a_3\varepsilon_1\varepsilon_2 B(\frac{\varepsilon_2}{2}, \frac{\varepsilon_2+2}{2})\sin\eta^{\varepsilon_1-1}\cos\eta^{2\varepsilon_1+1}(k_x\sin\eta^{\varepsilon_1}+1) \\ &\quad \cdot(k_y\sin\eta^{\varepsilon_1}+1)d\eta. \end{aligned} \tag{5.6}$$

Consequently, the volume of a tapered superquadric can be evaluated as

$$\begin{aligned} V_t &= 2\int_0^{a_3} A_t(z)dz = 2\int_0^{\pi/2} A_t(\eta)\dot{z}(\eta)d\eta \\ &= 4a_1a_2a_3\varepsilon_1\varepsilon_2 B(\frac{\varepsilon_2}{2}, \frac{\varepsilon_2+2}{2})\int_0^{\pi/2}(k_x\sin\eta^{\varepsilon_1}+1)(k_y\sin\eta^{\varepsilon_1}+1)\sin\eta^{\varepsilon_1-1} \\ &\quad \cdot\cos\eta^{2\varepsilon_1+1}d\eta \\ &= 2a_1a_2a_3\varepsilon_1\varepsilon_2 B(\frac{\varepsilon_2}{2}, \frac{\varepsilon_2+2}{2})[B(\frac{\varepsilon_1}{2}, \varepsilon_1+1) + k_xk_yB\left(\frac{3\varepsilon_1}{2}, \varepsilon_1+1\right) \\ &\quad +(k_x+k_y)B(\varepsilon_1, \varepsilon_1+1)]. \end{aligned} \tag{5.7}$$

After obtaining the volume of a tapered superquadric, the object vector i_i^L of can be evaluated by Equation (5.2). The second-order inertial moments of a tapered superquadric can be derived in a similar manner. The superquadric is first cut along the z axis into slices of infinitesimal thickness dz parallel to the xy plane, then using Steiner's rule, the inertial moments of a superellipse about the x, y and z axes are evaluated as

$$\begin{aligned} (I_{xx}^0)_t &= \int\int_S y^2dxdy = f_y^3f_xI_{xx}^0 = f_y^3f_x\frac{1}{2}ab^3\varepsilon B(\frac{3\varepsilon}{2}, \frac{\varepsilon}{2}), \\ (I_{yy}^0)_t &= \int\int_S x^2dxdy = f_x^3f_yI_{yy}^0 = f_x^3f_y\frac{1}{2}a^3b\varepsilon B(\frac{\varepsilon}{2}, \frac{3\varepsilon}{2}), \\ (I_{zz}^0)_t &= \int\int_S (x^2+y^2)dxdy = (I_{xx}^0)_t + (I_{yy}^0)_t \\ &= f_y^3f_x\frac{1}{2}ab^3\varepsilon B(\frac{3\varepsilon}{2}, \frac{\varepsilon}{2}) + f_x^3f_y\frac{1}{2}a^3b\varepsilon B(\frac{\varepsilon}{2}, \frac{3\varepsilon}{2}), \end{aligned} \tag{5.8}$$

where f_x and f_y are tapering transformations along the x and y axes as shown in Equation (2.24). Consequently, the inertial moments $(I_{xx})_t$ of a tapered superquadric along the x axis

can be derived as

$$(I_{xx})_t = \int\int\int_V (y^2+z^2)dxdydz = \int_{-a_3}^{a_3}[(I_{xx}{}^0)_t + z^2 A_t]dz$$
$$= \frac{1}{2}a_1a_2a_3\varepsilon_1\varepsilon_2[a_2^2(Ixx)_1 + 4a_3^2 B(\frac{\varepsilon_2}{2},\frac{\varepsilon_2}{2}+1)B\left(\frac{3\varepsilon_1}{2},\varepsilon_1+1\right)], \quad (5.9)$$

where

$$(I_{xx})_1 = \int_{-a_3}^{a_3}(I_{xx}{}^0(z))_t dz = \int_{-\pi/2}^{\pi/2}(I_{xx}{}^0(\eta))_t \dot{z}(\eta)d\eta$$
$$= \int_{-\pi/2}^{\pi/2} f_y^3 f_x I_{xx}^0 a_3\varepsilon_1 \sin\eta^{\varepsilon_1-1}\cos\eta d\eta$$
$$= k_x k_y^3 B(\frac{5\varepsilon_1}{2},2\varepsilon_1+1) + (3k_xk_y^2+k_y^3)B(2\varepsilon_1,2\varepsilon_1+1)$$
$$+(3k_xk_y+3k_y^2)B(\frac{3\varepsilon_1}{2},2\varepsilon_1+1) + (k_x+3k_y)B(\varepsilon_1,2\varepsilon_1+1). \quad (5.10)$$

Similarly, the inertial moments $(I_{yy})_t$ of a tapered superquadrics along the y axis can be derived as

$$(I_{yy})_t = \int\int\int_V (x^2+z^2)dxdydz$$
$$= \frac{1}{2}a_1a_2a_3\varepsilon_1\varepsilon_2[a_1^2(I_{yy})_1 + 4a_3^2 B(\frac{\varepsilon_2}{2},\frac{\varepsilon_2}{2}+1)B(\frac{3\varepsilon_1}{2},\varepsilon_1+1)], \quad (5.11)$$

where

$$(I_{yy})_1 = k_x^3 k_y B(\frac{5\varepsilon_1}{2},2\varepsilon_1+1) + (3k_x^2k_y+k_x^3)B(2\varepsilon_1,2\varepsilon_1+1)$$
$$+(3k_xk_y+3k_x^2)B(\frac{3\varepsilon_1}{2},2\varepsilon_1+1) + (3k_x+k_y)B(\varepsilon_1,2\varepsilon_1+1). \quad (5.12)$$

The inertial moments $(I_{zz})_t$ of a tapered superquadrics along the z axis can therefore be derived as

$$(I_{zz})_t = \int\int\int_V (x^2+y^2)dxdydz = \int_{-\pi/2}^{\pi/2}(I_{zz}^0(\eta))_t\dot{z}(\eta)d\eta$$
$$= \frac{1}{2}a_1a_2a_3\varepsilon_1\varepsilon_2(a_2^2(I_{xx})_1 + a_1^2(I_{yy})_1). \quad (5.13)$$

5.2 The View Registration Algorithm

Most image registration techniques are based on raw images and require a certain amount of overlap to successfully establish correspondences between views. For this reason, the computational complexity of this type of image registration algorithm is usually very high due to

the extensive searching and the optimization involved. For our multi-view superquadric representation algorithm, superquadrics have been recovered for each single-view data before the multi-view data are registered. Therefore, we want to utilize recovered superquadrics as much as possible in the subsequent steps of the multi-view superquadric representation algorithm. For this reason, we improve and extend the range image registration algorithm (Jaklič, 1997) and present a 3D view registration algorithm based on superquadrics.

We extend the algorithm (Jaklič, 1997) in both technical and application aspects. From the technical point of view, we extend the original algorithm to register tapered as well as regular superquadrics by deriving the first- and second- order inertial moments for a tapered superquadric. Second, we extend the initial 2D range image registration algorithm to 3D view registration. From the application point of view, the proposed view registration algorithm can register scenes containing complex backgrounds and multiple objects. Not all the objects need to be represented by superquadrics precisely. The only assumption of the proposed view registration algorithm is that there should be at least one object which can be represented by superquadrics in the scene, and this object is visible from the views to be registered. As long as this condition is satisfied, the proposed approach can register arbitrary scenes. This condition can be easily satisfied for most real indoor or outdoor scenes. A typical example can be found in a hazardous environment clean-up application (National Energy Technology Laboratory, deactivition & decommissioning focus area, 2002).

In summary, the proposed view registration approach extends and improves the 2D range image registration approach (Jaklič, 1997) in the following aspects.

- We derive the first- and second- order inertial moments for tapered superquadrics.
- We extend the initial 2D range image registration approach to register 3D unstructured data sets.
- Our algorithm can register more generic and complicated scenes consisting of complex backgrounds and multiple objects.

The proposed view registration algorithm based on superquadric representation has two major advantages over other surface registration methods. The first advantage results from the nature of superquadrics. In other words, since the proposed registration technique relies on recovered superquadrics, which are volumetric primitives, and can be successfully recovered from single-view, incomplete data sets, this technique requires neither a time-consuming point correspondence searching nor a large amount of overlap between two views while most existing registration methods require at least 20% overlap (Chua and Jarvis, 1996; Roth, 1999). The second advantage is that since superquadrics can be successfully recovered from only unstructured point clouds without face information, no triangulation step is needed. Therefore, the proposed algorithm is free from the mesh resolution inconsistency problem which needs to be handled by an extra mesh uniform step in other registration approaches (Chen and Medioni, 1992; Johnson and Hebert, 1998).

Given a pair of 3D data sets obtained from two views to be registered, superquadrics are first recovered from each view. A new coordinate frame is then constructed for each view, which is rigidly attached to the object in each view. After the two frames are constructed,

their relationships to the world coordinate system can be calculated. Consequently, the rigid transformation between the two frames is estimated, and the two views are registered. The major step-by-step procedures involved are summarized in the following algorithm followed by a detailed description.

Algorithm 4 (3D View Registration)

Input: Two 3D unstructured data sets in the format of (x, y, z) obtained from two views.

Step 1. Recover superquadrics from each view to be registered. Select superquadrics recovered for both views to be used for view registration.

Step 2. Compute the first- and second-order inertial moments of each selected superquadric in its canonical coordinate system.

Step 3. Transform the inertial moments into the world coordinate system.

Step 4. For each view, sum up the inertial moments of the recovered superquadrics and construct a new canonical coordinate frame.

Step 5. Evaluate the rigid transformation between the two views.

Output: A homogeneous matrix representing the rigid transformation between the two views.

A step-by-step description of the proposed view registration algorithm is provided as follows.

Step 1. Recover superquadrics from each view. Within our multi-view superquadric representation algorithm, multi-part objects have been decomposed into single parts before they are fitted to superquadrics. Therefore, superquadric recovery of single-part objects is a straightforward data fitting problem, and can be performed using the method presented in Section 3.4. In addition, superquadrics can be recovered using other methods such as the recover-and-select algorithm (Leonardis et al., 1997). Our view registration algorithm has no constraint on superquadric representation methods.

Step 2. Compute inertial moments of superquadrics. The first- and second-order inertial moments of selected superquadrics are evaluated in their canonical coordinate systems. We are able to evaluate inertial moments of both regular and tapered superquadrics. The evaluation of regular and tapered superquadrics are presented in Sections 2.5 and 5.1.

Step 3. Transform the moments into the world coordinate system. The inertial moments obtained in Step 2 are in the canonical coordinate system of each superquadric. They need to be transformed into the global (world) coordinate system. Transformation of the object vector i is easy to implement. We use superscripts G and L to present global and canonical coordinate systems. The object vector i in the global system is obtained by

$$i^G = \begin{bmatrix} I_x \\ I_y \\ I_z \\ V \end{bmatrix} = \begin{bmatrix} \int_V x dV \\ \int_V y dV \\ \int_V z dV \\ \int_V dV \end{bmatrix} = T \begin{bmatrix} I_x^L \\ I_y^L \\ I_z^L \\ V^L \end{bmatrix} = T i^L, \tag{5.14}$$

where T is the homogeneous transformation matrix including the translation and the rotation shown in Equation (2.12).

Transformation of the object matrix I is much more difficult. The transformation needs to be decomposed into translation and rotation first. The object matrix is next transformed under translation and rotation respectively. Matrix T_t and T_r represent translation and rotation, the same as in Equation (2.11). Under translation, the second-order moments are converted to

$$
\begin{aligned}
I_{xx}^G &= \int_V (y^2+z^2)dV = I_{xx}^L + 2p_y I_y^L + 2p_z I_z^L + (p_y^2+p_z^2)V \\
I_{yy}^G &= \int_V (x^2+z^2)dV = I_{yy}^L + 2p_x I_x^L + 2p_z I_z^L + (p_x^2+p_z^2)V \\
I_{zz}^G &= \int_V (x^2+y^2)dV = I_{zz}^L + 2p_x I_x^L + 2p_y I_y^L + (p_x^2+p_y^2)V \\
I_{xy}^G &= -D_{xy}^G = -\int_V xydV = -\left(D_{xy}^L + p_x I_y^L + p_y I_x^L + p_x p_y V\right) \\
I_{xz}^G &= -D_{xz}^G = -\int_V xzdV = -\left(D_{xz}^L + p_x I_z^L + p_z I_x^L + p_x p_z V\right) \\
I_{yz}^G &= -D_{yz}^G = -\int_V yzdV = -\left(D_{yz}^L + p_y I_z^L + p_z I_y^L + p_y p_z V\right).
\end{aligned} \tag{5.15}
$$

Since $I_x^L = I_y^L = I_z^L = 0$ and $I_{xy}^L = I_{yz}^L = I_{xz}^L = 0$ in the canonical system, the above equations can be simplified to

$$
\begin{aligned}
I_{xx}^G &= \int_V (y^2+z^2)dV = I_{xx}^L + (p_y^2+p_z^2)V \\
I_{yy}^G &= \int_V (x^2+z^2)dV = I_{yy}^L + (p_x^2+p_z^2)V \\
I_{zz}^G &= \int_V (x^2+y^2)dV = I_{zz}^L + (p_x^2+p_y^2)V \\
I_{xy}^G &= -D_{xy}^G = -\int_V xydV = -p_x p_y V \\
I_{xz}^G &= -D_{xz}^G = -\int_V xzdV = -p_x p_z V \\
I_{yz}^G &= -D_{yz}^G = -\int_V yzdV = -p_y p_z V.
\end{aligned} \tag{5.16}
$$

The translation of the object matrix I can be denoted by

$$
I^G = translation(T_t, I^L, i^L). \tag{5.17}
$$

Under rotation, the simple rule holds as

$$
I^G = T_{rot-lu} I^L T_{rot-lu}^{-1}, \tag{5.18}
$$

where matrix T_{rot-lu} is the left-up 3×3 matrix of T_r. Therefore, the object matrix I is converted into the world coordinate system as

$$I^G = T_{rot-lu} translation(T_t, I^L, i^L) T_{rot-lu}^{-1}. \tag{5.19}$$

Step 4. Construct a new canonical coordinate system for each view. In this step, the object vector i and matrix I of superquadrics are first summed up. The new canonical coordinate systems for the two views are denoted by C_1 and C_2. To construct a new canonical frame, translation and rotation between the global and the new canonical system needs to be evaluated separately. To recover the translation, a new frame C is constructed in which the following equation holds

$$i_C = \begin{bmatrix} 0 \\ 0 \\ 0 \\ V \end{bmatrix} = (T_G^{C1})_{translation} i_G = \begin{bmatrix} 1 & 0 & 0 & p_x \\ 0 & 1 & 0 & p_y \\ 0 & 0 & 1 & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} I_x \\ I_y \\ I_z \\ V \end{bmatrix}. \tag{5.20}$$

Therefore, the translation becomes

$$(T_G^C)_t = \begin{bmatrix} 1 & 0 & 0 & -I_x/V \\ 0 & 1 & 0 & -I_y/V \\ 0 & 0 & 1 & -I_z/V \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{5.21}$$

The object matrix in this new frame C can be evaluated by Equation (5.17) as

$$I_C = translation((T_G^C)_t, I, i). \tag{5.22}$$

Next, the rotation matrix rotates frame C to final constructed frame C_1 in such a way that

$$I_{C_1} = \begin{bmatrix} I_{xx}^{C_1} & 0 & 0 \\ 0 & I_{yy}^{C_1} & 0 \\ 0 & 0 & I_{zz}^{C_1} \end{bmatrix} = T_{rot-lu} I_C (T_{rot-lu})^{-1}. \tag{5.23}$$

Since the matrix I_C is a real symmetric matrix, it can be diagonalized by an orthogonal matrix Q

$$I_{C_1} = Q^{-1} I_C Q, \tag{5.24}$$

where the columns of Q contain a complete set of orthonormal eigenvectors of I_C. Therefore, the rotation matrix is derived and given by

$$T_{rot-lu} = Q^{-1}. \tag{5.25}$$

Thus, the transformation between the newly constructed canonical frame C_1 and the global system is

$$T_G^{C_1} = T_r (T_G^C)_t, \tag{5.26}$$

where T_r is the homogeneous matrix of its left-up 3×3 rotation matrix T_{rot-lu}.

Step 5. Evaluate the rigid transformation between the two views. Since a canonical frame is rigidly attached to an object, the same point in the same object expressed in various canonical systems should have the same coordinates. Based on this rule, the transformation between two views can be evaluated. Terms C_1 and C_2 represent two canonical systems for two views, and G_1 and G_2 represent two corresponding global systems. Matrix T_{C1}^{G1}, T_{C2}^{G2} are transformations from a canonical coordinate system (C_1, C_2) to a global coordinate system (G_1, G_2). For a point X on the object surface, the following relationships hold

$$X_{G_1} = T_{C_1}^{G_1} X_{C_1}, \quad X_{G_2} = T_{C_2}^{G_2} X_{C_2}, \quad \text{and} \quad X_{C_1} = X_{C_2}, \tag{5.27}$$

where X_{C_1} represents the point X in the canonical coordinate system C_1, and X_{G_1} represents the point X in the global coordinate system G_1. Therefore, the rigid transformation between the two views can be derived as

$$T = T_{C_2}^{G_2} (T_{C_1}^{G_1})^{-1}. \tag{5.28}$$

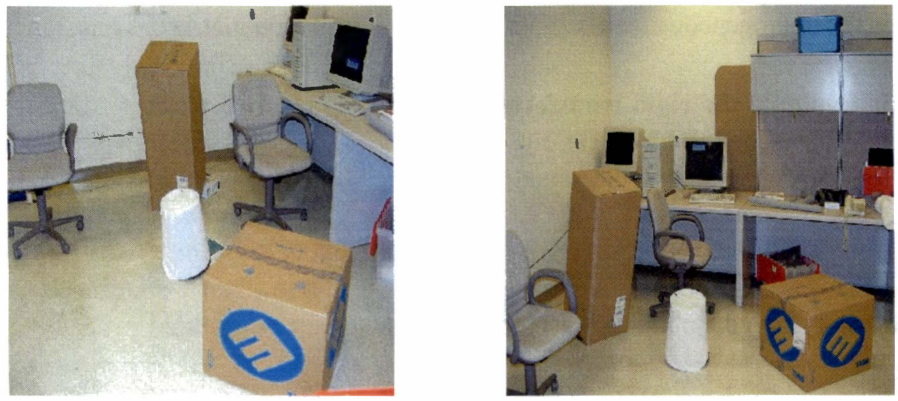
Since the inertial moments are invariant to rotation of the coordinate frame for 180^0 about any coordinate axes, there are four possible orientations for the constructed canonical system: the constructed one itself and one for each rotation of 180^0 about the x, y and z axes, respectively (Jaklič, 1997). Among the four transformations, the one with the minimum registration error is selected as correct registration between the two views. This process is explained as follows.

We assume that superquadric descriptions D_{11} and D_{12} in the first data set are obtained from view 1, and descriptions D_{21} and D_{22} in the second data set are obtained from view 2. D_{ij} consists of region R_{ij} and superquadric model M_{ij} $(i, j = 1, 2)$. The following delineates the correct transformation and the corresponding descriptions between the two images. The four transformations, T_1, T_2, T_3, and T_4, have been calculated. For every transformation T_i, we first keep the representation of the first view unchanged and transform the representation in the second view into the first view. Next, we calculate distances between the superquadric models in the first view and the transformed regions from the second view. The registration error E_1 for the first view is evaluated as

$$\begin{aligned} E_1 &= min(E_{11}, E_{12}) + min(E_{13}, E_{14}) \\ E_{11} &= \sum_{x \in T(R_{21})} d(x, M_{11}), \quad E_{12} = \sum_{x \in T(R_{22})} d(x, M_{11}) \\ E_{13} &= \sum_{x \in T(R_{21})} d(x, M_{12}), \quad E_{14} = \sum_{x \in T(R_{22})} d(x, M_{12}), \end{aligned} \tag{5.29}$$

where $d(x, M_{ij})$ represents the distance between point x in the region R and model M_{ij}.

Similarly, The registration error E_2 for the second view can be evaluated by keeping the representation of the second data set unchanged, and transforming the representation in the first



(a) (b)

Figure 5.1: An office scene scanned from two views. The range images are acquired by the RIEGL scanner. Superquadrics are recovered for the two boxes and the barrel on the floor and used in view registration. (a) Photograph of original scene from view 1 and (b) photograph of original scene from view 2.

data set into the second data set. E_2 is evaluated as

$$\begin{aligned} E_2 &= min(E_{21}, E_{22}) + min(E_{23}, E_{24}) \\ E_{21} &= \sum_{y \in T(R_{11})} d(y, M_{21}), \quad E_{22} = \sum_{y \in T(R_{12})} d(y, M_{21}) \\ E_{23} &= \sum_{y \in T(R_{11})} d(y, M_{22}), \quad E_{24} = \sum_{y \in T(R_{12})} d(y, M_{22}). \end{aligned} \tag{5.30}$$

Therefore, two registration errors, E_1 and E_2, are obtained for each transformation. For the correct transformation, E_1 should be very close to E_2, and both should be less than a threshold E_τ. With this method, the correct transformation T between the two views can be identified.

The proposed view registration algorithm can also register complex scenes consisting of background and multiple objects obtained from two viewpoints. This algorithm extends superquadric representation to address the generic 3D data registration problem. For instance, for an office scene scanned from two views shown in Fig. 5.1, superquadrics can easily be recovered for the tall box, the short box, and the barrel on the floor for each view. Our view registration algorithm based on superquadrics can then register the two views. Similar to other surface registration techniques, the registration result obtained by the proposed method can be further refined by the iterative closest point (ICP) registration algorithm (Besl and McKay, 1992; Zhang, 1994).

5.3 Summary

Due to the fact that the mapping from an object to the corresponding recovered superquadric is one-to-one, we apply superquadrics to the generic 3D view registration problem and present a view registration algorithm based on superquadrics. Unlike most image or surface registration techniques, our approach does not need points or feature correspondences, and the registration result (rigid transformation) can be directly obtained from the recovered superquadrics. A comparison between the proposed view registration algorithm and those existing feature correspondence-based 3D surface registration algorithms is illustrated in Fig. 5.2. Based on the diagrams shown in Fig. 5.2, we observe that the proposed registration algorithm, which is based on superquadric representation, has two major advantages over other correspondence-based registration algorithms. The first advantage is that the proposed method does not need to establish feature correspondences between two views, a task which requires a huge amount of computation. In other words, the proposed method does not need a large amount of overlap between two views, while this overlap is required in most correspondence-based algorithms (Chua and Jarvis, 1996; Roth, 1999). The second advantage is that the proposed method can register unstructured 3D data while most other methods need triangle meshes as input. This advantage eliminates the mesh resolution inconsistency problem that needs to be handled by an extra mesh uniform step (Johnson and Hebert, 1998; Chen and Medioni, 1992).

Occlusion has been an unsolved problem for 3D surface registration. The occlusion handling capability of the proposed registration method depends solely on the pre-segmentation step, i.e., whether single objects can be successfully segmented from a scene containing occlusions. Once objects are segmented from the scene, the remaining scene data, whether there are occlusions or not, do not affect the performance of the view registration. The proposed view registration method is able to directly evaluate the rigid transformation between two views from the segmented objects (3D unstructured data) and associated superquadrics. In contrast, most steps of the correspondence-based registration methods including feature extraction and correspondence identification are subject to occlusions. To be specific, false features could be extracted from a scene containing occlusions. Primitives evaluated based on feature points tend to be inaccurate or even wrong in the presence of occlusions. Furthermore, it is extremely difficult to establish correct primitive correspondences when occlusions occur. Therefore, we conclude that the proposed view registration method has better occlusion handling abilities.

The only assumption required by our approach is that at least one object in the scene should be representable by superquadrics. This condition can be easily satisfied in many cases. In addition, the proposed view registration approach does not require a time-consuming correspondence search nor much overlap between views. Furthermore, since superquadrics can be successfully recovered from only 3D points without face information, no triangulation step is needed, which eliminates the mesh resolution inconsistency problem. Therefore, the proposed registration approach has lower computational cost since only a few superquadric parameters are involved in the registration evaluation.

In summary, the advantages of the extended view registration algorithm include the abilities to

- Handle unstructured 3D data, i.e., point clouds without mesh information,

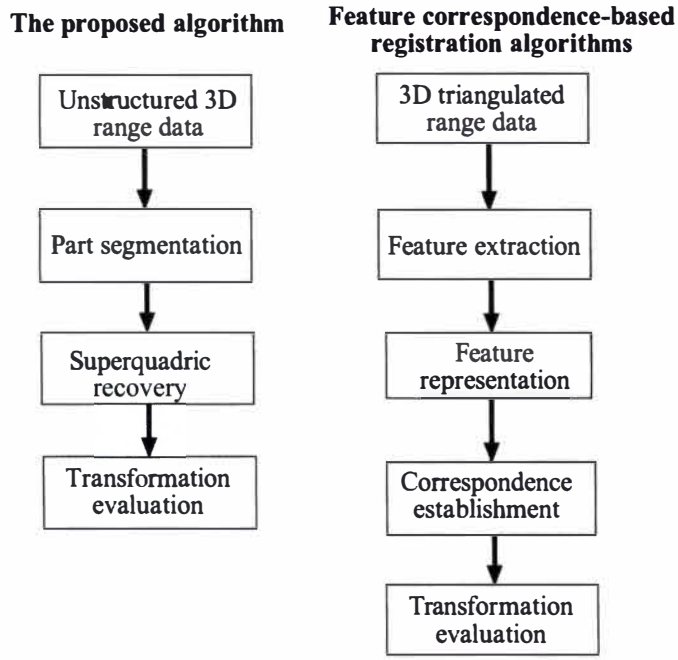


Figure 5.2: Diagrams of the registration algorithms.

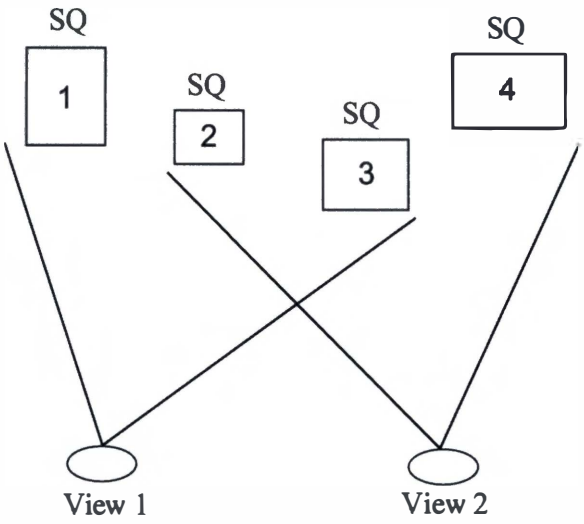


Figure 5.3: View registration based on superquadrics.

- Register tapered superquadrics, and
- Register pre-segmented complex scenes.

The limitations of the view registration algorithm are that at least one object in the scene must be representable by superquadrics, and that each superquadric is visible from views to be registered. For instance, for a scene from which four superquadrics are recovered as shown in Fig. 5.3, superquadric "1", "2", "3" are visible from the first view and superquadric "2", "3", "4" are visible from the second view. Since only superquadric "2" and "3" are visible from both views, either of the two superquadrics or both of them can be used to register the scene. The more superquadrics used, the higher accuracy can be achieved with higher computational cost.

This chapter presented an extended and improved 3D view registration algorithm and its application to scenes. Experimental results on 3D surface registration of real scenes utilizing superquadrics are shown in Section 7.2. The proposed view registration algorithm based on superquadric representation has been published in (Zhang et al., 2001; Zhang et al., 2002a).

Chapter 6

Experimental Results I: Supplementary Background

This chapter presents experimental results on several supplementary issues. Although we do not claim contributions on these issues, they are crucial to succeeding experiments in this dissertation. For instance, several different objective functions have been used in the literature of superquadric representation. However, no convincing conclusion about their performances has been derived from the literature. Therefore, we present a comprehensive performance evaluation of two primarily used objective functions in this chapter. Section 6.1 presents a brief introduction of two range scanners used in this dissertation including the triangulation-based, IVP range sensor and the time-of-flight, RIEGL laser scanner. Experiments on rendering and visualization of 3D bent superquadrics from our proposed quadrant analysis technique are provided in Section 6.2. Section 6.3 presents a comprehensive comparison between two objective functions for superquadric representation. Finally, Section 6.4 summarizes this chapter.

6.1 Range Scanners

Range Scanners measure distances between object surfaces and the scanner itself, and collect 3D coordinate data from object surfaces. Range images explicitly represent 3D surface geometry in a sampled form. Range imaging techniques have been evolving for over thirty years and have found more and more applications in areas including manufacturing industry, robotic navigation, medical diagnosis, etc. Generally speaking, current range scanners can be classified into active and passive types. In the former case, a special lighting device illuminates the scene, and in the latter case, scene illumination is provided by the ambient light. Active sensors are used when the accuracy of the captured data is a major concern and the scene itself cannot provide sufficient illumination. A taxonomy of various range sensing techniques is shown in Fig. 6.1. More detailed surveys of range imaging techniques can be found in (Besl, 1989; Nitzan, 1989).

Among the active sensors, the two most popular range scanners are time-of-flight and triangulation-based. The time-of-flight scanners have a laser diode that sends a pulsed laser beam to the scanned object (Nitzan, 1989). The pulse is diffusely reflected by the surface, and part of the light returns to the receiver. The time required for the light to travel from the laser

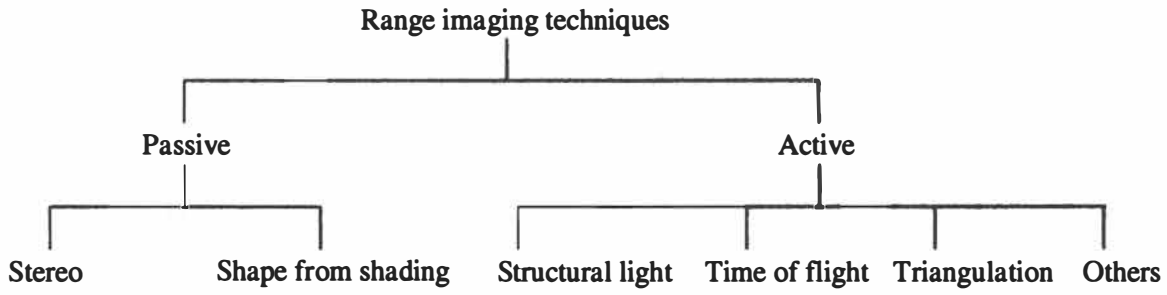


Figure 6.1: A taxonomy of various range imaging techniques.

diode to the object surface and back is measured, and the distance between the laser and the object surface can therefore be calculated using the speed of light. The scanning principle for time-of-light range scanners is depicted in Fig. 6.2(a). This type of scanner is able to measure larger distances than triangulation-based scanners. On the other hand, they are less accurate, especially at a close range. The accuracy of this kind of scanner is subject to the distance between the object and the scanner.

For the triangulation-based scanners, a light spot or stripe is projected onto an object surface, and the position of the spot on the object is recorded by one or more CCD cameras (Nitzan, 1989). The angle of the light beam leaving the scanner is internally recorded, and the fixed base length between the laser source and the camera is known from calibration. The distance between the object surface to the camera is geometrically calculated from the recorded angle and the base length as shown in Fig. 6.2(b). The accuracy of this kind of scanner is subject to both the base length of the scanner and the distance between the scanner and the object. This type of scanner is usually used to measure small objects within a short range.

In our experiments, time-of-flight and triangulation-based scanners are used in different cases. The triangulation-based scanner is used to measure small objects while the time-of-flight scanner is used to scan large scenes. The first range imaging system is the IVP Ranger Scanner System (Integrated Vision Products, 2000), which contains a triangulation-based sheet-of-light laser rangefinder as shown in Fig. 6.3(a). Due to the sheet-of-light utilized, the range camera only finds one range profile of the scene at each exposure. A scanning mechanism is needed to obtain a 2D range image. A horizontal conveyor belt is used to move objects to be scanned at a user-selected speed as shown in Fig. 6.3(a). Appendix A provides a comprehensive characterization of the Ranger scanner. When the distance between the objects and the scanner is larger than 5 m, or the objects cannot be put onto the conveyor belt such as an office scene, we use the time-of-flight based RIEGL scanner (RIEGL Laser Measurement Systems, 2000) shown in Fig. 6.3(b) to scan the objects or the scene.

6.2 Visualization and Recovery of Bent Superquadrics in 3D

Based on the quadrant analysis technique proposed in Section 3.4, a variety of bent superquadrics are rendered from pre-specified parameters and shown in the following figures. Fig. 6.4 shows

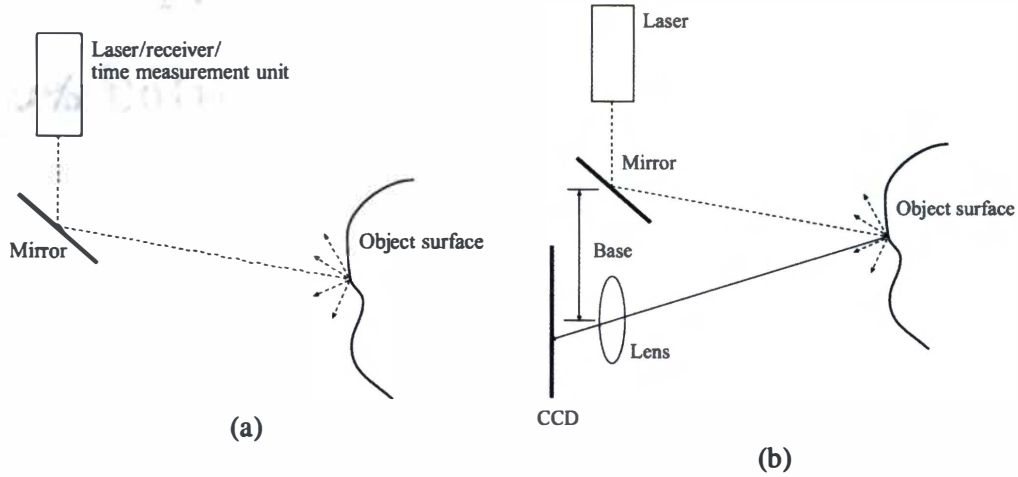


Figure 6.2: Scanning principles of two types of range scanners. (a) Time-of-flight and (b) triangulation.

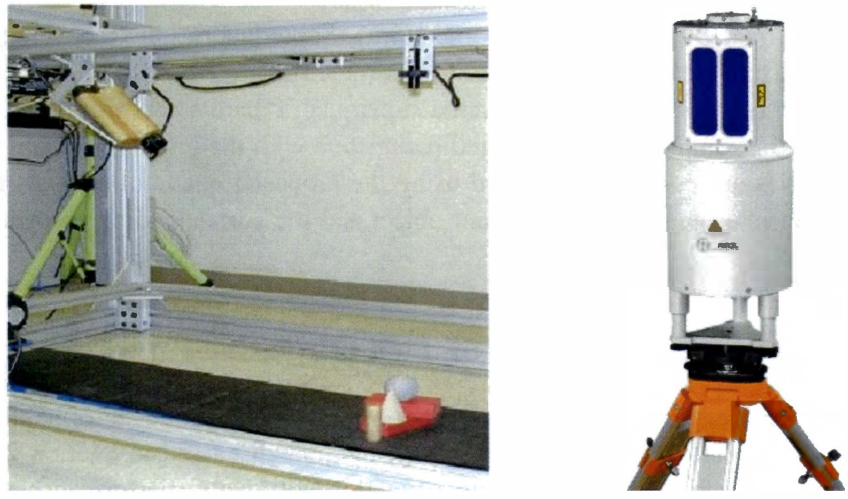
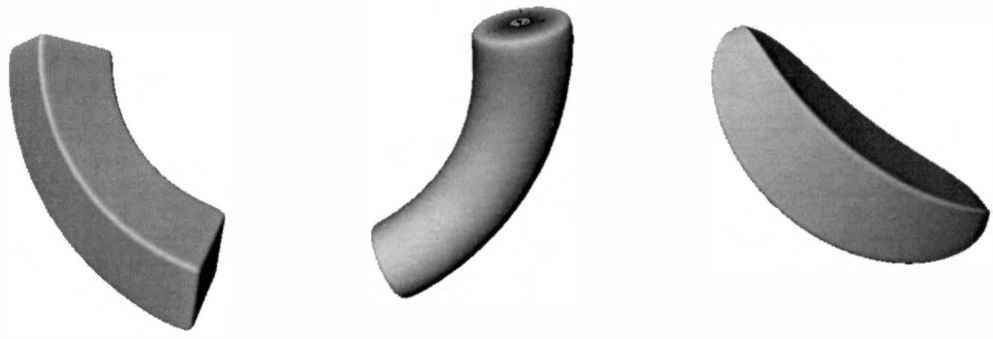


Figure 6.3: Triangulation-based and time-of-flight range scanners. (a) Triangulation-based scanner and (b) time-of-flight, long-range, RIEGL 3D imaging sensor.



(a) (b) (c)

Figure 6.4: Visualization of 3D bent superquadrics. These bent superquadrics have various shape parameters. The remaining superquadric parameters are $(20, 20, 70, 0.01, 1.0)$ for $(a_1, a_2, a_3, k, \alpha)$. With pre-specified parameters, superquadrics are generated from their explicit definition and rendered in 3D using quad meshes. (a) $\varepsilon_1 = 0.1, \varepsilon_2 = 0.1$, (b) $\varepsilon_1 = 0.1, \varepsilon_2 = 1.0$, and (c) $\varepsilon_1 = 1.0, \varepsilon_2 = 0.1$.

three bent superquadrics with different shape parameters. Fig. 6.5 shows tapered and bent superquadrics. From these figures, it has been found that the proposed quadrant analysis technique is able to render bent superquadrics in arbitrary 3D space successfully.

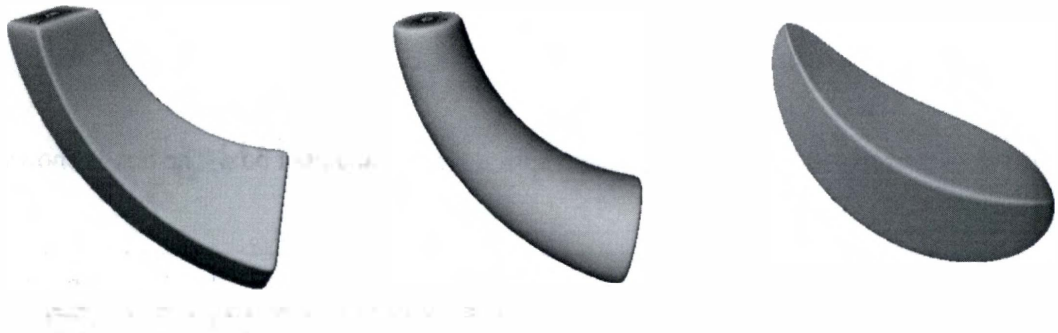
Correspondingly, with our quadrant analysis technique, superquadrics can be successfully recovered from the synthetic data shown in these figures. The unstructured 3D points shown in Fig. 6.6 are extracted from Figs. 6.4(b) and 6.5(a). Superquadrics are recovered from these point clouds. The parameters used to create the superquadrics are ground truths for superquadric recovery. Tables. 6.1 and 6.2 show recovered superquadric parameters from the point clouds shown in Fig. 6.6. Comparing the recovered parameters with the ground truths, it is seen that bent superquadrics are correctly recovered using the proposed quadrant analysis technique. Fig. 6.7 shows a real range image of a bent object and the corresponding recovered bent superquadric model.

6.3 Performance Evaluation of Two Objective Functions

This section presents experimental results on recovering both regular and deformable superquadrics from single-part objects using the two objective functions discussed in Section 2.5.4 and investigates their performances. The objective function with better performance will be used throughout this dissertation.

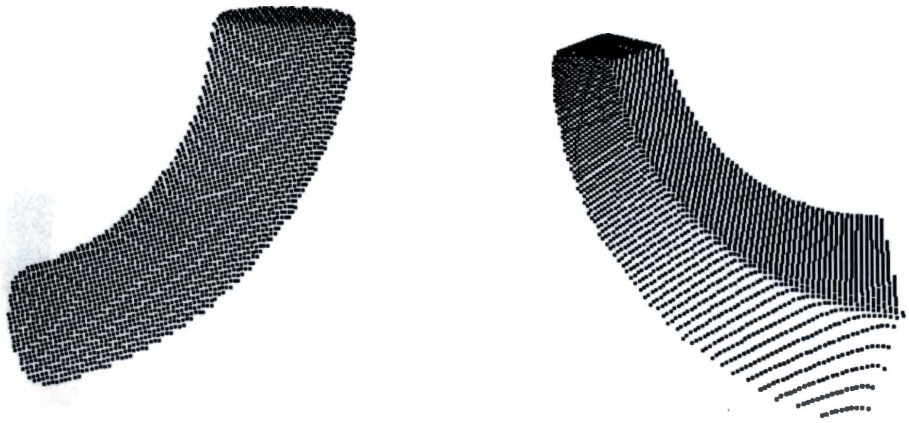
The two objective functions are expressed as

$$G_1(\Lambda) = a_1 a_2 a_3 \sum_{i=1}^{N} (F^{\varepsilon_1}(x_c, y_c, z_c) - 1)^2, \tag{6.1}$$



(a) (b) (c)

Figure 6.5: Visualization of 3D bent and tapered superquadrics. These superquadrics have various shape parameters. The remaining superquadric parameters are $(20, 20, 70, 0.3, 0.3, 0.01, 1.0)$ for $(a_1, a_2, a_3, k_x, k_y, k, \alpha)$. With pre-specified parameters, superquadrics are generated from their explicit definition and rendered in 3D using quad meshes. (a) $\varepsilon_1 = 0.1, \varepsilon_2 = 0.1$, (b) $\varepsilon_1 = 0.1, \varepsilon_2 = 1.0$, and (c) $\varepsilon_1 = 1.0, \varepsilon_2 = 0.1$.



(a) (b)

Figure 6.6: 3D unstructured point clouds used in superquadric representation. (a) 3D point clouds extracted from Fig. 6.4(b) and (b) 3D point clouds extracted from Fig. 6.5(a).

Table 6.1: Recovered superquadric parameters for a bent cylinder. The cylinder is shown in Fig. 6.6(a).

	a_1	a_2	a_3	ε_1	ε_2	k	α
Ground truth	20	20	70	0.1	1.0	0.01	1.0
Parameter	18.79	19.23	67.92	0.09	1.05	0.009	0.95

Table 6.2: Recovered superquadric parameters for a tapered and bent box. The box is shown in Fig. 6.6(b).

View	a_1	a_2	a_3	ε_1	ε_2	k_x	k_y	k	α
Ground truth	20	20	70	0.1	0.1	0.3	0.3	0.01	1.0
Parameter	19.25	17.41	68.56	0.10	0.08	0.29	0.27	0.009	0.97

(a) (b)

Figure 6.7: Superquadric representation results for a bent object. This mesh is a single-view scan from the IVP Ranger Scanner System. With recovered parameters, superquadrics are generated from their explicit definition and rendered in 3D using quad meshes. (a) Rendering of original mesh and (b) rendering of recovered superquadric.

and

$$G_2(\Lambda) = \sum_{i=1}^{N} d^2 = \sum_{i=1}^{N} (|r_0||1 - F^{-\frac{\epsilon_1}{2}}(x_c, y_c, z_c)|)^2. \tag{6.2}$$

Another objective function G_0 used for discussion is expressed as

$$G_0(\Lambda) = \sum_{i=1}^{N} (F^{\epsilon_1}(x_c, y_c, z_c) - 1)^2. \tag{6.3}$$

In the experiments, both real and synthetic range data are tested. Experimental results on regular and deformable superquadrics are shown and discussed respectively. Robustness against noise and sensitivity to different viewpoints are explored as well. The accuracy, fitting errors of the recovered superquadrics and the convergence speed of the data fitting process are compared for the two objective functions.

For superquadric fitting, 30 iterations are used in the Lavenberg-Marquardt algorithm for each data set. The fitting error between raw data and recovered superquadrics is calculated as

$$e = \frac{1}{N} \sum_{i=1}^{N} d_i^2, \tag{6.4}$$

where N represents the number of points. Variable d_i represents the radial Euclidean distance between the ith point of the input data and the corresponding recovered superquadric surface.

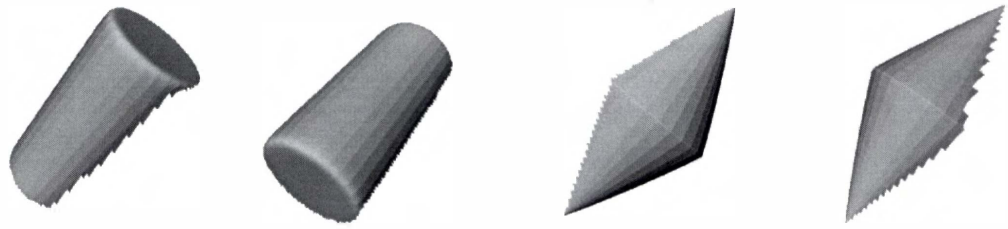
6.3.1 Synthetic Data with and without Noise

This section classifies experimental results with synthetic range data into three different categories as regular superquadrics with and without noise, and deformable superquadrics without noise. The superquadric models are created using ground truth parameters. Multiple single-view range data are captured from different viewpoints, and used to investigate the effect of varying viewpoints on the two objective functions.

Regular Superquadrics without Noise

Several experiments on recovering regular superquadric models from noise-free synthetic range data are conducted. Fig. 6.8 shows two sets of 3D range data for a cylinder and a bi-directional cone collected from two different views. Table 6.3 indicates the ground truth values used to create the cylinders and the cones shown in Fig. 6.8 and summarizes recovered superquadric parameters for the objects using the two objective functions given in Equations (2.35) and (2.36).

Comparing the recovered superquadric parameters with the ground truth values in Table 6.3, for the range data of the cylinder shown in Fig. 6.8, the superquadric parameters recovered using the objective function based on radial Euclidean distance (Function 2) are slightly closer to the ground truth values than those using the objective function based on the implicit definition of superquadrics (Function 1). Despite the slight difference in fitting errors, the two objective functions provide almost the same representation results. The volume of the cylinder shown in



(a) (b) (c) (d)

Figure 6.8: Synthetic range data extracted from superquadrics. The two superquadrics include a cylinder and a bi-directional cone acquired from two views. (a) 3D data extracted from the cylinder from view 1 consisting of 12,763 points, (b) 3D data obtained from view 2 consisting of 12,038 points, (c) 3D data extracted from the bi-directional cone from view 1 consisting of 11,384 points, and (d) 3D data obtained from view 2 consisting of 11,257 points.

Table 6.3: Recovered superquadric parameters for a cylinder and a bi-directional cone. The data are scanned from two views as shown in Fig. 6.8. *Objective function 1* is based on the implicit definition as shown in Equation (2.35). *Objective function 2* is based on the radial Euclidean distance as shown in Equation (2.36). The fitting error is calculated from Equation (6.4).

Object	Data		a_1	a_2	a_3	ε_1	ε_2	Fitting Error
Cylinder	Ground truth		30	30	60	0.1	1.0	
	Objective Function 1	View 1	28.58	25.58	56.50	0.1	1.01	0.67
		View 2	25.23	28.73	57.50	0.1	1.04	0.72
	Objective Function 2	View 1	29.24	27.19	58.379	0.1	1.02	0.35
		View 2	28.86	29.61	58.81	0.1	1.01	0.31
Cone	Ground truth		30	30	60	1.9	1.0	
	Objective Function 1	View 1	22.79	25.58	49.01	1.66	0.87	1.21
		View 2	24.33	27.08	51.49	1.75	0.92	1.08
	Objective Function 2	View 1	28.37	29.81	57.88	1.90	1.01	0.43
		View 2	28.36	23.58	59.38	1.88	1.07	0.64

Fig. 6.8 can be represented as $Ca_1a_2a_3$, where C is a constant. Therefore, the final recovered superquadric model with minimum volume also has a minimum value in Function 1. The biased fitting caused by objective function G_0 defined in Equation (6.3) is completely compensated by the term $a_1a_2a_3$. The two objective functions exhibit similar performances in this experiment.

The other set of noise-free, synthetic range data extracted from a bi-directional cone is shown in Figs. 6.8(c) and 6.8(d) with ground truth parameters given in Table 6.3. As shown in Table 6.3, there are significant differences in both fitting errors and recovered parameters for the two objective functions. In this case, the volume of the object not only depends on $a_1a_2a_3$, but on ε_1 and ε_2. Therefore, the term $a_1a_2a_3$ in Function 1 fails to provide the minimum volume restriction to the recovered model. Consequently, the biased fitting caused by G_0 given in Equation (6.3) is not completely compensated by the additional term $a_1a_2a_3$. This is why Function 1 recovers inaccurate superquadrics in this case. Function 2 performs consistently well because geometric distance is an accurate measurement of closeness between raw data and fitted models for all shapes.

Regular Superquadrics with Additive Noise

In order to estimate the effect of noise, simulated random noises ranging from 1.0% to 10.0% are added to the cylinder shown in Fig. 6.8(a). The drand48() function is used in the program to generate the random noise. This function uses a linear congruential algorithm and 48-bit integer arithmetic to generate a nonnegative double uniformly distributed over the interval [0.0, 1.0] (Press et al., 1992). Table 6.4 shows corresponding recovered superquadric parameters for these data with varying noise levels. The corresponding noisy versions of the figure are shown in Fig. 6.9. From the 3D data sets with various noise levels shown in Fig. 6.9, and corresponding recovered superquadric parameters illustrated in Table 6.4, it can be observed that in all cases, objective function 2 recovers better superquadric parameters compared with ground truths. It can also be seen that objective function 1 recovers unacceptable superquadric parameters for the data with 6.0% random noise while objective function 2 recovers superquadric parameters with acceptable accuracy even for the data with 10.0% random noise in terms of shape parameters ε_1 and ε_2.

Likewise, random Gaussian noise with the standard deviation at 1.0% is added to the data shown in Figs. 6.8(c) and 6.8(d). The corresponding noisy versions of the synthetic data is shown in Fig. 6.10. Table 6.5 shows corresponding ground truths and recovered superquadric parameters. Comparing the recovered parameters in Table 6.5 to the ground truth values, it is seen that for the range data in Fig. 6.10, the parameters recovered using Function 2 are significantly better than those using Function 1. The fitting errors show the same tendency. Comparing the parameters recovered from both noisy and noise-free data, we see that Function 2 is more robust to noise and viewpoints than Function 1.

Deformable Superquadrics without Noise

Experiments on recovering superquadrics with global deformations from noise-free synthetic range data are also conducted. Fig. 6.11 shows 3D range images of a tapered cylinder captured from two different views. Table 6.6 summarizes recovered superquadric parameters for this

Table 6.4: Recovered superquadric parameters for a cylinder. The original data have additive random noise at the level of 1% through 10% as shown in Figs. 6.9 (a) through (j).

Data	Function	a_1	a_2	a_3	ε_1	ε_2	Fitting Error
Ground truth		30	30	60	0.1	1.0	
1% noise	Function 1	27.67	29.51	58.83	0.1	1.02	1.44
	Function 2	27.89	29.61	59.11	0.1	1.02	1.32
2% noise	Function 1	28.09	29.60	58.55	0.1	1.0	1.42
	Function 2	28.56	29.97	59.22	0.1	1.01	1.30
3% noise	Function 1	28.72	29.62	58.17	0.1	0.98	1.52
	Function 2	29.69	30.45	59.25	0.11	1.01	1.42
4% noise	Function 1	29.59	29.60	57.83	0.1	0.93	1.38
	Function 2	30.91	31.35	59.26	0.11	0.98	1.21
5% noise	Function 1	30.66	29.60	57.58	0.1	0.89	2.14
	Function 2	31.56	33.45	59.34	0.11	0.97	2.05
6% noise	Function 1	31.86	29.61	57.41	0.12	0.84	2.87
	Function 2	32.24	35.66	59.48	0.11	0.96	2.65
7% noise	Function 1	33.06	29.64	57.31	0.1	0.79	4.46
	Function 2	32.92	31.90	59.68	0.13	0.94	3.64
8% noise	Function 1	34.17	29.68	57.24	0.1	0.75	5.37
	Function 2	33.86	30.02	59.91	0.13	0.94	4.42
9% noise	Function 1	35.08	29.74	57.20	0.1	0.71	6.28
	Function 2	34.74	32.05	60.18	0.13	0.94	5.14
10% noise	Function 1	35.70	29.80	57.16	0.1	0.67	7.52
	Function 2	33.25	32.50	60.51	0.14	0.91	6.64

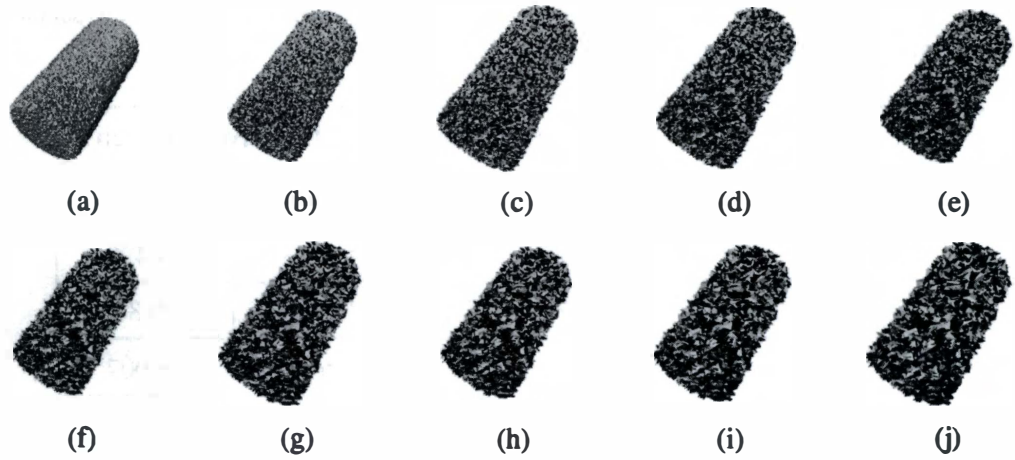


Figure 6.9: Synthetic 3D range data of a cylinder. Two-view scans are acquired with additive random noise ranging from (a) 1%, (b) 2%, (c) 3%, (d) 4%, (e) 5%, (f) 6%, (g) 7%, (h) 8%, (i) 9%, and (j) 10% .

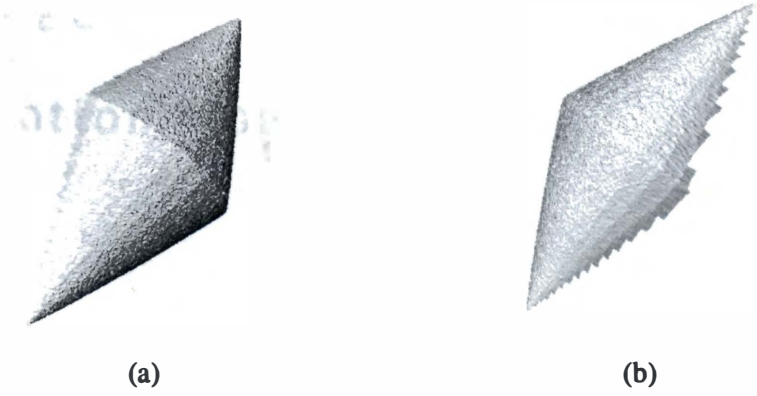
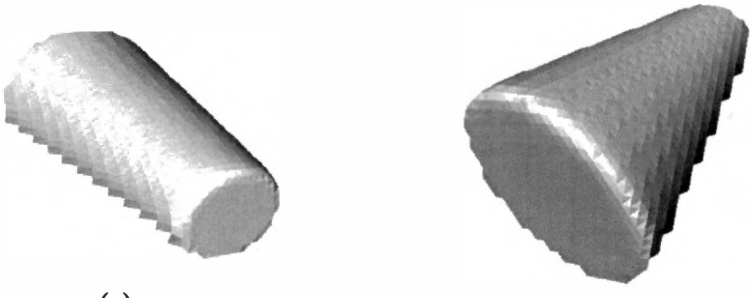


Figure 6.10: Synthetic range data of a bi-directional cone. Two-view scans are acquired with 1.0% random Gaussian noise.

Table 6.5: Recovered superquadric parameters for a bi-directional cone. Two-view scans with 1.0% random Gaussian noise as shown in Fig. 6.10.

		Objective Function 1		**Objective Function 2**	
Parameter	Ground truth	**View 1**	**View 2**	**View 1**	**View 2**
a_1	30	13.359	24.374	28.379	28.490
a_2	30	16.691	27.114	29.806	29.80
a_3	60	31.749	51.560	57.889	58.007
ε_1	1.9	0.395	1.752	1.896	1.895
ε_2	1.0	0.456	0.923	1.013	1.011
Fitting Error		8.331	2.042	0.62	0.55



(a) (b)

Figure 6.11: Synthetic range data of a tapered cylinder. Two-view scans are acquired. (a) 3D data obtained from view 1 consisting of 2,196 points and (b) 3D data obtained from view 2 consisting of 1,857 points.

Table 6.6: Recovered superquadric parameters for a tapered cylinder. Two-view scans are acquired as shown in Fig. 6.11.

		Objective Function 1		Objective Function 2	
Parameter	Ground truth	View 1	View 2	View 1	View 2
a_1	30	18.13	28.18	30.51	28.68
a_2	30	26.91	22.09	28.44	27.68
a_3	60	61.80	57.72	62.51	59.63
ε_1	0.1	0.12	0.15	0.1	0.11
ε_2	1.0	1.17	1.11	1.15	1.09
t_x	0.4	0.39	0.36	0.43	0.36
t_y	0.4	0.41	0.23	0.50	0.38
Fitting Error		5.395	2.032	1.241	1.812

tapered object using the two objective functions. From the recovered superquadric parameters shown in Table 6.6, we observe that for the range data in Fig. 6.11, the parameters recovered using Function 2 are significantly better than those using Function 1, compared with the ground truth values. The fitting error generated by Function 2 is much smaller than that generated by Function 1. Similar to the data shown in Fig. 6.8, the volume of the deformable model shown in Fig. 6.11 is subject not only to $a_1a_2a_3$, but to k_x and k_y. Therefore, the item $a_1a_2a_3$ in Function 1 failed to completely compensate the biased fitting caused by G_0. Function 2 does not suffer from this problem because it is purely geometric distance-based, and is insensitive to shape variations.

6.3.2 Real Range Data

This section presents the experimental results for superquadric recovery from real range data. Both regular and deformed superquadrics are explored. The real, single-view range images are captured from the IVP Ranger Scanner System (Integrated Vision Products, 2000) and calibrated into 3D data. The unit for the calibrated 3D data is mm.

Regular Superquadrics

A block and a cylinder with their measured ground truth parameters are used in the experiments. Fig. 6.12 shows the raw range data of the block and the cylinder. Table 6.7 shows recovered superquadric parameters for these two objects.

(a) (b) (c) (d)

Figure 6.12: Real range data for various objects. The obejcts include a block, a cylinder, a cone, and a bent object. The block and the cylinder can be represented by regular superquadrics while the cone and the bent object need to be represented by globally deformed superquadrics. The range images are scanned and calibrated from the IVP Ranger Scanner System. (a) Rendering of original mesh for a block, (b) rendering of original mesh for a cylinder, (c) rendering of original mesh for a cone and (d) rendering of original mesh for a bent object.

Table 6.7: Recovered superquadric parameters for three objects. The objects include a block, a cylinder, and a cone shown in Fig. 6.12. GT is the ground truth.

Object	Function	a_1	a_2	a_3	ε_1	ε_2	k_x	k_y	Fitting Error
Block	GT	29	56	116	0.1	0.1	0	0	
	Function 1	16.38	53.73	107.46	0.1	0.1	0	0	6.35
	Function 2	23.26	60.83	111.23	0.1	0.11	0	0	3.42
Cylinder	GT	52	52	50	0.1	1.0	0	0	
	Function 1	35.75	35.09	51.56	0.1	0.70	0	0	6.13
	Function 2	51.79	53.73	51.82	0.1	1.0	0	0	1.34
Cone	GT	48	48	75	0.1	1.0	0.9	0.9	
	Function 1	23.73	32.01	90.98	0.1	1.12	0.83	0.72	9.40
	Function 2	50.39	46.45	78.78	0.1	1.08	0.85	0.60	2.40

From Table 6.7, we observe that for both the cylinder and the block, the recovered superquadric parameters using function 2 are obviously much better than those using Function 1. The fitting error generated by Function 2 is also much smaller than that generated by Function 1. The reason that Function 1 yields poor results in these cases is similar to the reason in the case of synthetic data with noise. In such cases, the term $a_1a_2a_3$ in Function 1 failed to compensate the biased fitting caused by G_0 in the presence of noise.

Deformable Superquadrics

A cone and a bent object are explored in the experiments to investigate how the two objective functions perform on the recovery of deformable superquadrics from real range data. Figs. 6.12(c) and 6.12(d) show calibrated 3D range data for these objects. Table 6.7 summarizes ground truth values and recovered superquadric parameters for the cone. Similarly, comparing the parameters recovered by the two objective functions with the ground truth values in Table 6.7, for the real range data shown in Fig. 6.12(c), the parameters recovered using Function 2 are much closer to the measured ground truth values than those using Function 1, as is the case with synthetic range data. The fitting error generated by Function 2 is much smaller than that generated by Function 1. The reason is again similar to the reason indicated in the case of synthetic data containing deformable superquadrics. The volume of deformable superquadrics is not only subject to $a_1a_2a_3$, but also to the other parameters listed in Equation (2.30). Therefore, $a_1a_2a_3$ in Function 1 failed to compensate the biased fitting caused by G_0 shown in Equation (6.3). The recovered superquadric parameters for the object in Fig. 6.12(d) shows a similar tendency.

6.3.3 Convergence Speed

We investigate the convergence speed of the two objective functions through the experiments on the objects in Figs. 6.8(a) and 6.8(c). The evaluation is performed by calculating the fitting error versus the number of iterations. In the experiments, 50 iterations are used in the Lavenberg-Marquardt algorithm. The single-view data in Figs. 6.8(a) and 6.8(c) are tested and the corresponding convergence plots are shown in Fig. 6.13. For the cylinder shown in Fig. 6.8(a), we observe that Function 2 converges slightly faster, and has slightly smaller fitting errors throughout the iterations, as shown in Fig. 6.13(a). For the bi-directional cone shown in Fig. 6.8(c), Function 2 has a much faster convergence speed, and much smaller fitting errors, as shown in Fig. 6.13(b). In summary, Function 2 stabilizes after only 8-10 iterations, while Function 1 needs 20 iterations to stabilize. Furthermore, fitting errors of Function 1 are always significantly larger than those of Function 2 even after 50 iterations.

6.4 Summary

This chapter has presented supplementary experimental results on several issues. First, we briefly introduced the two range scanners used in this dissertation. Next, we show how our proposed quadrant analysis technique is able to render and recover bent superquadrics in 3D space. Experiments were conducted to comprehensively evaluate the performance of the two objective functions that are most common in the superquadric literature. These experiments involved

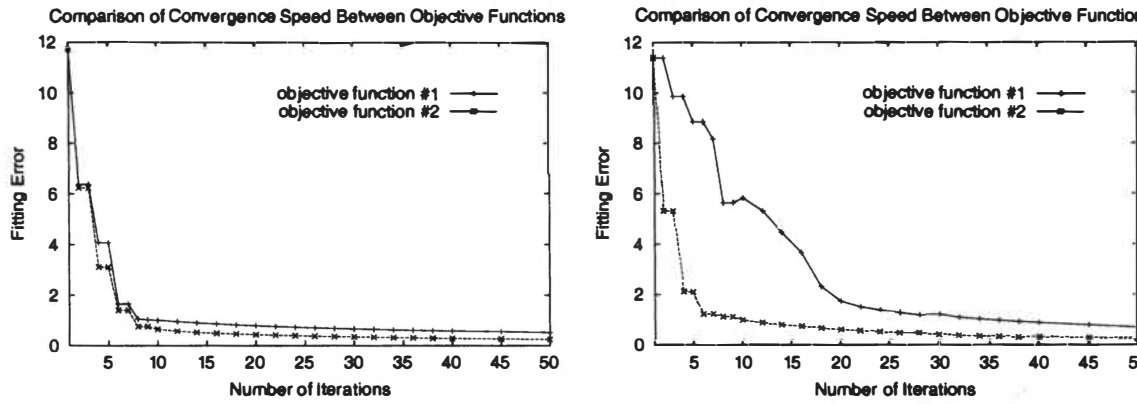


Figure 6.13: Comparison of convergence speed between two objective functions. (a) Convergence for the cylinder in Fig. 6.8(a) and (b) convergence for the bi-directional cone in Fig. 6.8(c).

regular and deformable suprequadrics, noisy and noise-free input data, different viewpoints, synthetic and real data, and convergence data. The objective function based on the Euclidean distance is demonstrated to have better overall performance in our experiments, and will be used throughout the dissertation.

Chapter 7

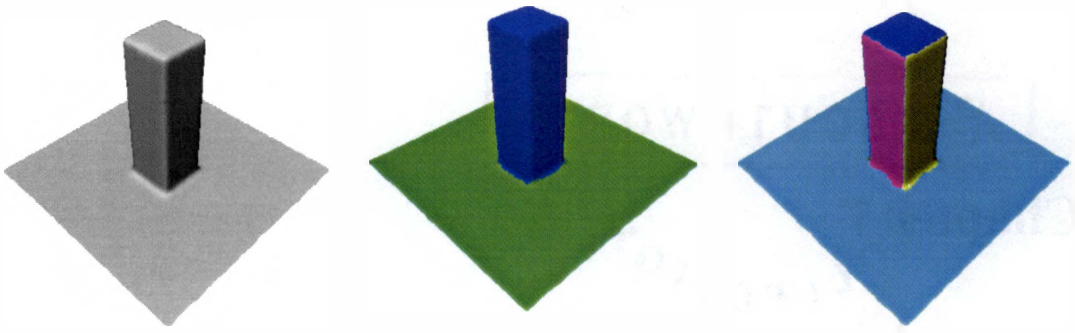
Experimental Results II: Contributions

This chapter presents experimental results for the algorithms proposed in this dissertation. We provide a thorough analysis of the strengths and limitations for each proposed algorithm. Among the three proposed contributions in this dissertation, the multi-view superquadric representation algorithm is essentially a complete pipeline from raw data to their superquadric representation. The other two contributions including the part decomposition algorithm and the view registration algorithm are two stages of the multi-view superquadric representation pipeline. This chapter first presents experimental results for each of these two stages and then for the whole pipeline.

We begin with results from the part decomposition algorithm in Section 7.1. This section demonstrates the overall capabilities of the algorithm through a large variety of synthetic and real data. Results on superquadric representation of multi-part objects and multi-object scenes utilizing part decomposition are presented in this section as well. In addition, we compare the performance of our superquadric representation approach utilizing part decomposition with the recover-and-select algorithm (Leonardis et al., 1997), which is the state of the art algorithm for superquadric representation. In Section 7.2, we investigate the capabilities of the view registration algorithm to demonstrate the improvements and extensions we make to the original algorithm. In Section 7.3, we complete the multi-view superquadric representation pipeline by presenting results for each step of the pipeline. In addition, we compare the performance of our multi-view superquadric representation approach with the recover-and-select algorithm. Finally, Section 7.4 concludes this chapter. This chapter demonstrates the efficiency of the proposed algorithms.

7.1 Part Decomposition and Superquadric Representation

The part decomposition algorithm decomposes objects into their constituent parts. Based on part decomposition, superquadric representation of multi-part objects can be performed in a two-step manner. The objects can first be decomposed into single parts. Next, each decomposed part is fitted to a superquadric model. Chapter 4 has described the part decomposition



(a) (b) (c)

Figure 7.1: Difference between part and region segmentation for a scene. The scene consists of two linked objects: a cube and a plane. The mesh is created from two superquadrics, and consists of 35,990 vertices 71,425 triangles. Part decomposition results consist of two parts. Region segmentation results consist of six regions: five regions for the cube and one region for the plane. (a) Rendering of original mesh, (b) part decomposition results, and (c) region segmentation results.

algorithm and the associated superquadric representation strategy in terms of motivations, step-by-step analysis, implementation issues, etc. The algorithms are implemented in C++ on a SGI Octane workstation. This section presents experimental results for the two algorithms. A wide variety of objects and scenes are used in the experiments. For part decomposition, we begin with a synthetic 3D model for a turret which is created by computer graphics techniques. Reconstructed 3D models for objects with simple geometries including a teapot and a mug follow. To demonstrate the application of our algorithms to automotive-related areas, reconstructed 3D models for automotive parts including a disc brake, a distributor cap, and a water neck are decomposed and represented by superquadrics. Next, single-view scans of a hammer, a dumbbell, a crossing-shaped pipe fitting, and a y-shaped pipe fitting are explored, which shows that our algorithms can handle single-view 3D meshes of common tools and industrial parts. Finally, a set of scenes consisting of multiple objects with various geometries and occlusions are investigated to simulate waste area clean-up and bin picking applications.

7.1.1 Part Decomposition of 3D Triangle Surfaces

Part decomposition classifies data points on object surfaces into groups, each of which corresponds to one physical part of the object. To make the goal of our proposed part decomposition algorithm clearer, Fig. 7.1 illustrates the difference between region segmentation and part decomposition. As illustrated in Fig. 7.1, a region segmentation algorithm segments the cylinder into three regions while the part decomposition segments the cylinder into one single part. Part decomposition is essential to high-level tasks such as superquadric representation and associated object recognition.

We have developed a part decomposition algorithm based on the human perception theory of transversality regularity. According to this theory, our algorithm identifies vertices of negative curvature minima as boundaries between two articulated parts. Next, each non-boundary vertex is grown into a part until it reaches the boundary. In this section, we analyze this algorithm through experimental results to find its strengths and weaknesses. Triangle meshes of a wide variety of objects and scenes are used as input throughout the experiments.

We begin with a synthetic 3D model for a turret which is created by computer graphics techniques. Reconstructed 3D models for objects with simple geometries including a teapot and a mug follow. To demonstrate the application of our part decomposition algorithm to automotive-related areas, reconstructed 3D models for automotive parts including a disc brake, a distributor cap, and a water neck are investigated. Next, single-view scans of a hammer, a dumbbell, a crossing-shaped pipe fitting, and a y-shaped pipe fitting are explored, which shows that our algorithm can handle single-view 3D meshes of common tools and industrial parts. Finally, a set of scenes consisting of multiple objects with various geometries and occlusions are investigated to simulate waste area clean-up and bin picking applications. The data we have explored in the experiments can be classified into the following categories.

- Synthetic and reconstructed 3D meshes for multi-part objects. A synthetic mesh is created by computer graphics techniques from a set of parameters and equations. A reconstructed mesh is obtained from multi-view range scans of a real object. A synthetic mesh of a turret and reconstructed meshes of objects including a teapot, a mug, a disc brake, a distributor cap, and a water neck are explored in the experiments.

- Automotive parts simulating applications in reverse engineering, CAD, etc. involving automotive objects. Reconstructed 3D meshes for a disc brake, a distributor cap, and a water neck are investigated to demonstrate the application of our part decomposition algorithm in these areas.

- Single-view range scans of a variety of objects and scenes containing multiple objects. We have included common tools and industrial objects such as a hammer, a dumbbell, a crossing-shaped pipe fitting, and a y-shaped pipe fitting. Scenes containing multiple objects with various geometries and occlusions are investigated to simulate waste area clean-up and bin picking applications. To be specific, the scenes contain a water neck, a computer mouse, a cup, a crossing-shaped pipe fitting, small bolts, power adapters, PVC pipes, and mechanical parts.

Table 7.1 provides a list of objects and scenes used in the experiments of part decomposition. This table briefly describes the objects regarding the number of vertices, the number of triangles, the number of constituent parts, and the computational time for part decomposition. We have two major sources for each mesh. We have either generated the meshes using our range scanners or used meshes from other laboratories or commercial companies. For meshes obtained in our laboratory, we have used either reconstructed surfaces or single-view scans of the objects. As shown in Table 7.1, the meshes for the teapot and the mug are reconstructed by Hoppe (Hoppe and Derose, 1992) at Microsoft Research. The mesh for the turret is created using computer graphics and downloaded from the online VRML gallery (VRML Model Gallery,

Table 7.1: List of objects and scenes for part decomposition.

Object Name	Figure Number	Number of Vertices	Number of Triangles	Number of Parts	Running Time (Seconds)
Turret	7.2	116	210	4	0.06
Teapot	7.3	3,034	6,010	5	3.4
Mug	7.4	1,725	3,450	2	0.97
Disc brake	7.5	37,171	73,394	2	443.91
Distributor cap	7.6	58,975	117,036	13	452.58
Water neck	7.7	58,784	117,564	9	307.99
Hammer	7.8	18,767	36,478	2	64.31
Dumbbell	7.9	11,610	22,542	3	26.58
"Cross" pipe fitting	7.10	17,963	35,238	4	41.47
"Y" pipe fitting	7.11	16,254	31,587	4	39.62
scene (I)	7.12	20,156	39,157	3	50.36
scene (II)	7.13	27,615	53,266	3	62.18
scene (III)	7.14	51,121	96,401	30	225.76
scene (IV)	7.15	90,237	175,298	18	546.58
scene (V)	7.16	89,561	169,479	49	407.58

(a) (b) (c)

Figure 7.2: Part decomposition results for a turret. This mesh is created by computer graphics and is downloaded from (VRML Model Gallery, OCNUS Company, 2002). The mesh consists of 116 vertices and 210 triangles while the decomposition consists of four parts. (a) Rendering of original mesh, (b) boundaries labeled in blue, and (c) decomposed parts labeled in different colors.

OCNUS Company, 2002). The remaining meshes are generated in our laboratory from the IVP Ranger System (Integrated Vision Products, 2000). Among them, the meshes for the disc brake, the distributor cap, and the water neck are reconstructions from multi-view scans (Sun et al., 2002). The meshes for the hammer, the dumbbell, the crossing-shaped pipe fitting, the y-shaped pipe fitting, and the five scenes containing miscellaneous objects are from single-view scans. By conducting experiments on both reconstructed and single-view meshes, we demonstrate the versatility of our part decomposition algorithm. Multi-part objects and multi-object scenes are explored separately in the following.

Multi-Part Objects

The following figures demonstrate part decomposition results for a variety of multi-part objects as listed in Table 7.1. Fig. 7.2 shows part decomposition results for a turret. The mesh for the turret is created using computer graphics techniques and is downloaded from (VRML Model Gallery, OCNUS Company, 2002). Fig. 7.2(b) demonstrates that the boundaries between two linked parts for the turret have been accurately identified without suffering the redundancy problem as with the teapot and the mug. This is attributed to the high quality of the original mesh. The mesh for the turret is directly generated from polyhedral shapes and, therefore, is noise free. Fig. 7.2(c) illustrates that the four visual parts are correctly decomposed for the turret, which is consistent with human perception.

Fig. 7.3 shows part decomposition results for a teapot. The mesh for the teapot is reconstructed by Hoppe (Hoppe and Derose, 1992) at Microsoft Research. Fig. 7.3(b) shows identified boundaries labeled blue. This figure demonstrates that vertices with negative Gaussian curvature compose boundaries between two articulated parts. In particular, we observe that the boundaries between the body and the cap, and the body and the small block at the bottom are

(a) (b) (c)

Figure 7.3: Part decomposition results for a teapot. This mesh is a reconstruction from Hughes Hoppe at Microsoft Research. The mesh consists of 3,042 vertices and 6,026 triangles while the decomposition consists of five parts. (a) Rendering of original mesh, (b) boundaries labeled in blue, and (c) decomposed parts labeled in different colors.

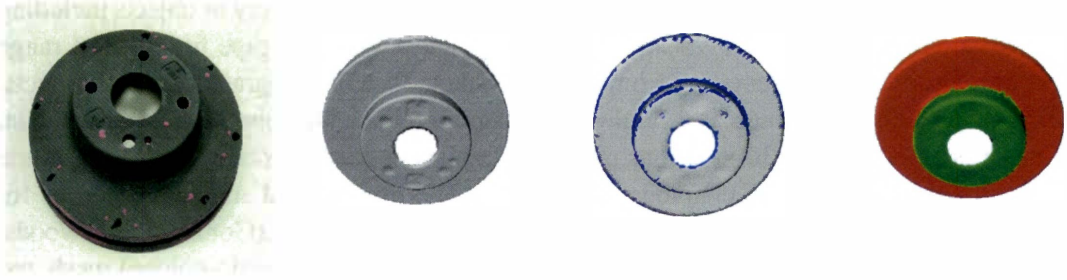
correctly identified. However, redundant boundaries between the body and the handle, and the body and the spout are identified. The rough surfaces on the spout and the handle in the original mesh as shown in Fig. 7.3(a) contributes to the inaccurate boundaries. This demonstrates that the resolution and the smoothness of the original mesh play important roles in curvature estimation and boundary detection. Fig. 7.3(c) indicates that the five single parts including the cap, the handle, the body, the spout, and the small block at the bottom are correctly decomposed for the teapot despite inaccurate boundaries. The part decomposition results for the teapot are consistent with human perception. The vertices on the spout which are not assigned to any single part can be further segmented by improving either the mesh resolution or the smoothness of the spout in the original mesh.

Fig. 7.4 shows part decomposition results for a mug. The mesh for the mug is reconstructed by Hoppe (Hoppe and Derose, 1992) at Microsoft Research. Fig. 7.4(b) indicates that the boundaries between the body and the handle are redundantly identified due to the low resolution of the original mesh. Fig. 7.4(c) shows that the two single parts including the body and the handle are correctly decomposed for the mug despite the redundantly identified boundaries on the handle. The part decomposition results for the mug using our algorithm are consistent with human perception.

The following three figures show part decomposition results for a variety of automotive parts including a disc brake, a distributor cap, and a water neck. The range images are acquired in our laboratory from the IVP Ranger System (Integrated Vision Products, 2000). The mesh is a reconstruction from multi-view scans (Sun et al., 2002). Fig. 7.5 shows part decomposition results for the disc brake. Fig. 7.5(c) indicates that redundant boundaries have been identified due to the rough surfaces of the original mesh. The smoothness of the reconstructed mesh is subject to factors such as the noise of the scanned range images, the registration and integration errors occurred in the reconstruction process, etc.

(a) (b) (c)

Figure 7.4: Part decomposition results for a mug. This mesh is a reconstruction from Hughes Hoppe at Microsoft Research. The mesh consists of 1,725 vertices and 3,450 triangles while the decomposition consists of two parts. (a) Rendering of original mesh, (b) boundaries labeled in blue, and (c) decomposed parts labeled in different colors.



(a) (b) (c) (d)

Figure 7.5: Part decomposition results for a disc brake. This mesh is a reconstruction from multiple range images using the IVP Ranger System. The mesh consists of 37,171 vertices and 73,394 triangles while the decomposition consists of two parts. (a) Photograph of original object, (b) rendering of original mesh, (c) boundaries labeled in blue, and (d) decomposed parts labeled in different colors.

The mesh cannot be infinitely smoothed to achieve perfect smoothness since we need to keep geometry details of the mesh as well. The more is the mesh smoothed, the less geometry details of the mesh can be kept. Therefore, in order to handle meshes reconstructed from real data, a part decomposition algorithm must be robust to the smoothness of the mesh to some extent. The part decomposition results for the teapot and the mug shown in Figs. 7.3 and 7.4 have demonstrated that our part decomposition algorithm can decompose objects correctly even if the corresponding meshes are not smooth. For the disc brake, the two decomposed cylindrical objects shown in Fig. 7.5(d) show this ability of our part decomposition algorithm. Furthermore, the part decomposition results for the disc brake demonstrate that our algorithm has reliable performance on meshes reconstructed from real range images.

Fig. 7.6 shows part decomposition results for the distributor cap. Fig. 7.6(c) shows that inaccurate boundaries have been identified between the tube on the top and the base of the distributor cap. This is attributed to factors including the low resolution, the rough surface on the tube, and the oversmoothed boundary between the tube and the base in the original mesh. Therefore, the tube fails to be decomposed as an individual part from the base. Despite this, Fig. 7.6(d) shows that most single parts including the five small cylinders, the base, two small screws, etc. are correctly decomposed for the distributor cap. This unideal decomposition indicates the limitation of our algorithm. The part decomposition results can be improved by increasing the resolution and the smoothness of the reconstructed mesh.

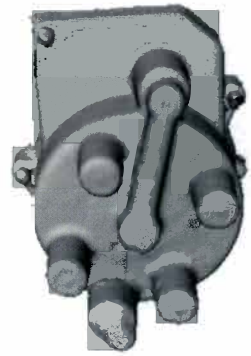
Fig. 7.7 shows part decomposition results for the water neck. Fig. 7.7(c) shows redundant boundaries on the handle of the water neck caused by the rough surfaces of the original mesh. However, as shown in Fig. 7.7(d), all the single parts including the cylindrical handle, the ball, the base, the small cylinder next to the handle, and several small screws have been correctly decomposed for the water neck despite the redundantly identified boundaries. This demonstrates that in most cases, our algorithm can handle triangle meshes in various smoothnesses and resolutions well.

The following four figures show part decomposition results for a variety of objects including a hammer, a dumbbell, a crossing-shaped pipe fitting, and a y-shaped pipe fitting. The range images are acquired in our laboratory from the IVP Ranger System (Integrated Vision Products, 2000). The mesh for each object is a single-view scan. To reduce the noise of the original data, each mesh is smoothed for 30 iterations using the Laplacian mesh smoothing function in the RapidForm software (INUS Technology, Inc, 2001). The original and smoothed meshes for the two pipe fittings are pseudo-colored using the method described in (Gonzalez and Woods, 2001) to illustrate the range depth variations on the surfaces. For the pseudo-colored mesh, red corresponds to smaller distances between the sensor and the surface of the object while blue corresponds to larger distances.

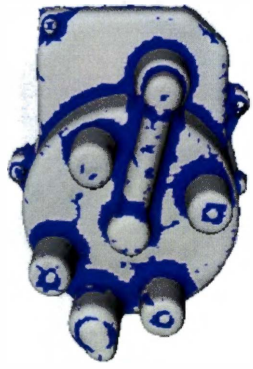
Figs. 7.8 and 7.9 show part decomposition results for the hammer and the dumbbell. As shown in Figs. 7.8(c) and 7.9(c), two visual parts are decomposed for the hammer and three for the dumbbell, which are consistent with human perception.

Fig. 7.10 shows part decomposition results for the $2in \times 2in \times 2in \times 2in$ crossing-shaped PVC (Poly Vinyl Chloride) pipe fitting. Compared with the gray-level meshes shown in Figs. 7.10(b) and 7.10(d), the pseudo-colored meshes shown in Figs. 7.10(c) and 7.10(e) illustrate more clearly the improvement of the mesh smoothness after Laplacian smoothing. Fig. 7.10(f)

(a)



(b)



(c)

(d)

Figure 7.6: Part decomposition results for a distributor cap. This mesh is a reconstruction from multiple range images using the IVP Ranger System. The mesh consists of 58,975 vertices and 117,036 triangles while the decomposition consists of 13 parts. (a) Photograph of original object, (b) rendering of original mesh, (c) boundaries labeled in blue, and (d) decomposed parts labeled in different colors.

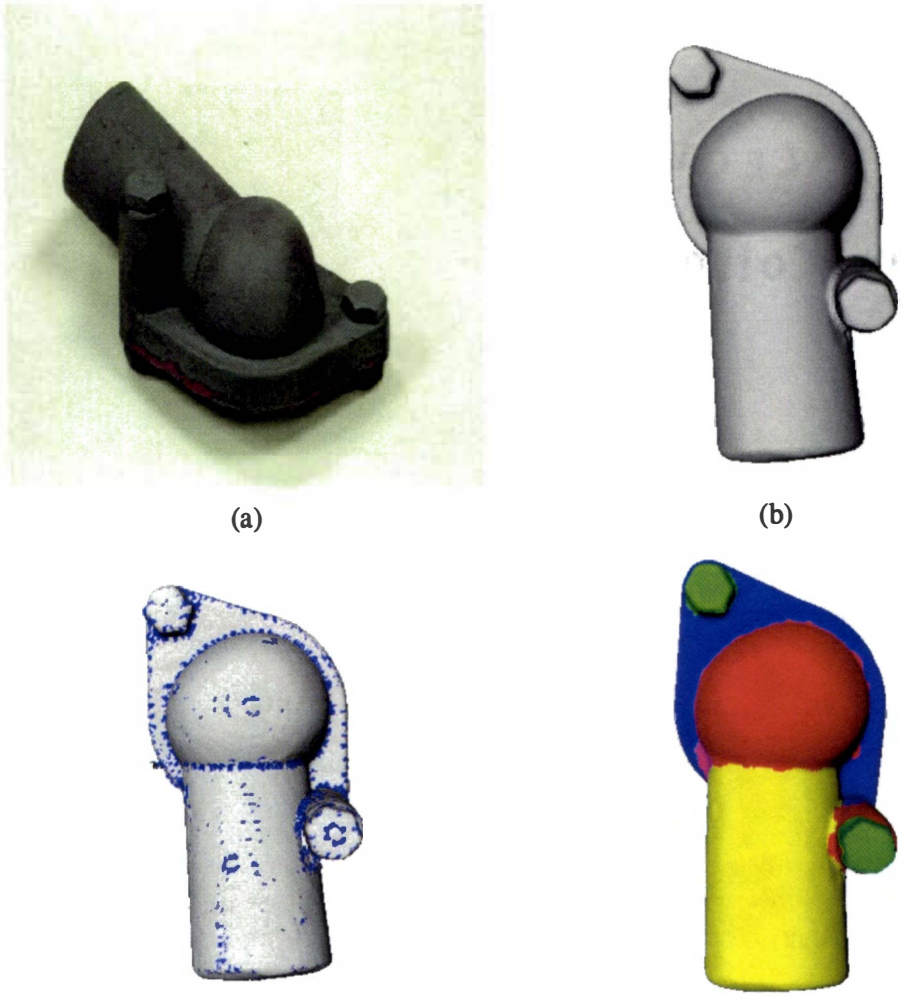


Figure 7.7: Part decomposition results for a water neck. This mesh is a reconstruction from multiple range images using the IVP Ranger System. The mesh consists of 58,784 vertices and 117,564 triangles while the decomposition consists of nine parts. (a) Photograph of original object, (b) rendering of original mesh, (c) boundaries labeled in blue, and (d) decomposed parts labeled in different colors.

(a) (b) (c)

Figure 7.8: Part decomposition results for a hammer. This mesh is a single-view scan from the IVP Ranger System. The mesh consists of 18,767 vertices and 36,478 triangles while the decomposition consists of two parts. To reduce the noise contained in the original data, the mesh is smoothed for 30 iterations using the Laplacian mesh smoothing function included in the RapidForm software. The smoothed data is then segmented. (a) Photograph of original object, (b) rendering of original mesh, and (c) decomposed parts labeled in different colors.

(a) (b) (c)

Figure 7.9: Part decomposition results for a dumbbell. This mesh is a single-view scan from the IVP Ranger System. The mesh consists of 11,610 vertices and 22,542 triangles while the decomposition consists of three parts. To reduce the noise contained in the original data, the mesh is smoothed for 30 iterations using the Laplacian mesh smoothing function included in the RapidForm software. The smoothed data is then segmented. (a) Photograph of original object, (b) rendering of original mesh, and (c) decomposed parts labeled in different colors.

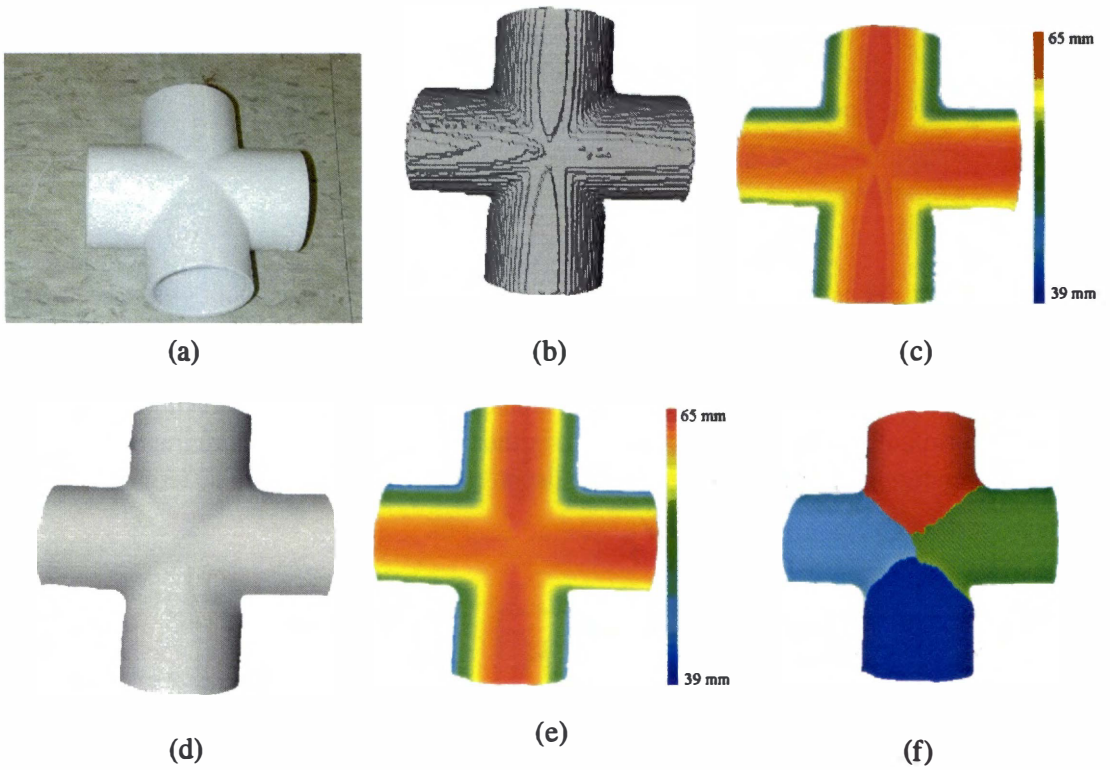


Figure 7.10: Part decomposition results for a crossing-shaped pipe fitting. The size of the pipe fitting is $2in \times 2in \times 2in \times 2in$. This mesh is a single-view scan from the IVP Ranger System. The mesh consists of 17,963 vertices and 35,238 triangles while the decomposition consists of four parts. To reduce the noise contained in the original data, the mesh is smoothed for 30 iterations using the Laplacian mesh smoothing function included in the RapidForm software. Both original and smoothed mesh are pseudo-colored to illustrate the range depth variations on the surfaces. Red corresponds to smaller distances between the sensor and the object surface while blue corresponds to larger distances. The smoothed data is then segmented. (a) Photograph of original object, (b) rendering of original mesh, (c) pseudo-colored data for the original mesh, (d) rendering of smoothed mesh, (e) pseudo-colored data for the smoothed mesh, and (f) decomposed parts labeled in different colors.

shows four decomposed cylinders for the crossing-shaped pipe fitting, which is consistent with human perception. The correct decomposition results demonstrate that our part decomposition algorithm can handle not only meshes reconstructed from multi-view scans, but also single-view triangle meshes.

Fig. 7.11 shows part decomposition results for the $2in \times 2in \times 2in$ y-shaped PVC (Poly Vinyl Chloride) pipe fitting. From Fig. 7.11(f), we observe that the y-shaped pipe fitting is decomposed into four single parts instead of five consistent with human perception. The two bigger cylinders at the center of the pipe fitting fail to be decomposed from each other due to the fuzzy boundary between them.

Scenes Containing Multiple Objects

The following five figures demonstrate part decomposition results for a variety of scenes containing multiple objects. The scenes consist of miscellaneous mechanical parts simulating bin picking applications. The range images are acquired and triangulated in our laboratory from the IVP Ranger System (Integrated Vision Products, 2000). Each mesh is a single-view scan. To reduce noise in the original scan, the mesh for each scene is smoothed for 30 iterations using the Laplacian mesh smoothing function included in the RapidForm software (INUS Technology, Inc, 2001).

Fig. 7.12 shows part decomposition results for a scene (I). This scene contains two separate blocks in order to investigate whether our part decomposition algorithm can handle isolated objects. Fig. 7.12(c) shows three decomposed parts including the two blocks and the background. The part decomposition results demonstrate that our algorithm can handle both connected and separated objects well.

Fig. 7.13 shows part decomposition results for a scene (II). To explore objects with various geometries, this scene contains a block (planar surfaces) and a sphere (cylindrical surfaces). Fig. 7.13(c) shows three decomposed parts including the block, the sphere, and the background. The part decomposition results demonstrate that our algorithm can handle scenes containing multiple, isolated objects of various shapes.

The original meshes for the scenes shown in the following three figures are pseudo-colored to illustrate the range depth variations on the surfaces. For the pseudo-colored mesh, red corresponds to smaller distances between the sensor and the surface of the object while blue corresponds to larger distances. Fig. 7.14 shows part decomposition results for a scene (III). This scene contains a larger variety of shapes than the previous scenes. The miscellaneous objects contained in this scene include five blocks, two spheres, three cylinders, three cones, a water neck, and a distributor cap simulating a bin picking application. Some objects in this scene are isolated and some are linked to each other. Since this scan is from a single view, the mesh leaves many objects as occluded and/or isolated and thus unconnected to other objects. Fig. 7.14(d) shows that 30 parts have been successfully decomposed for the scene. Especially, the handle and the ball of the water neck, and several single parts for the distributor cap have been correctly decomposed. The part decomposition results demonstrate that our algorithm can handle scenes containing a large variety of shapes with occlusions.

Fig. 7.15 shows part decomposition results for a scene (IV). This scene contains different objects from previous scenes. While the previous scenes mainly contain model-like objects,

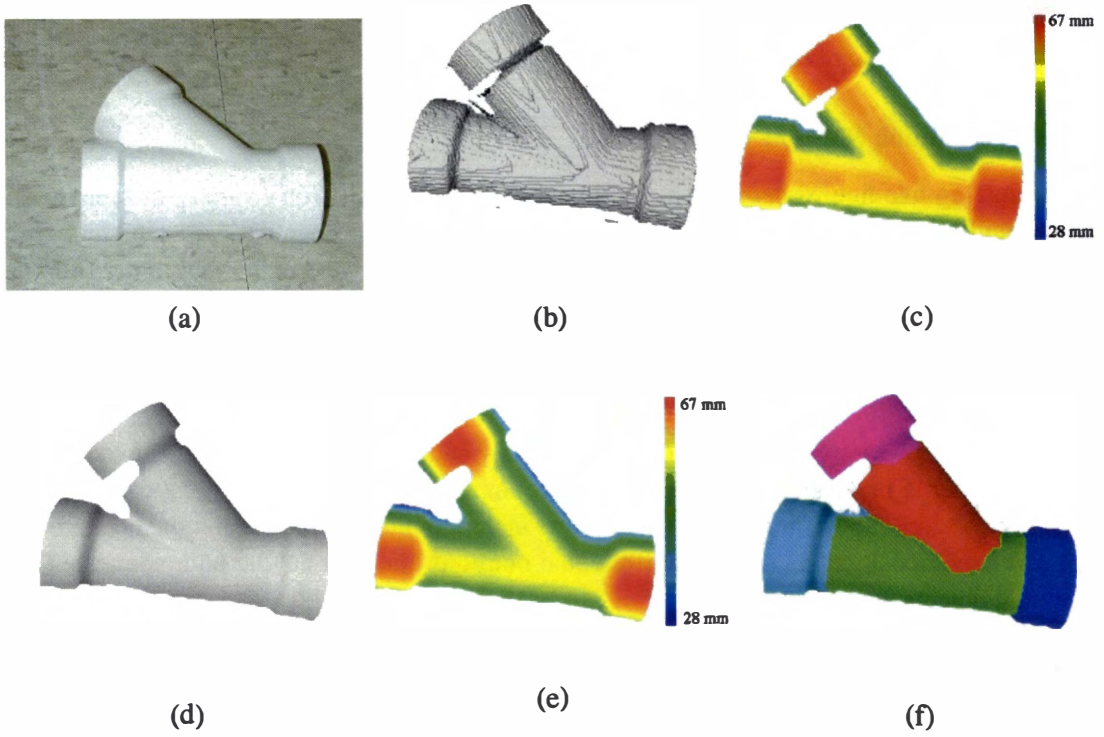
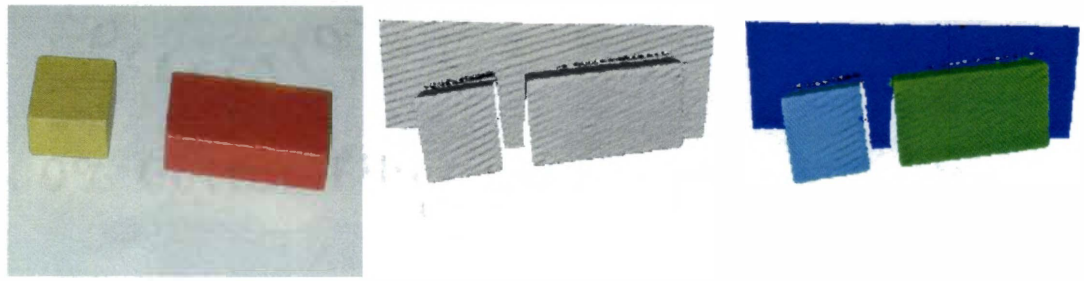
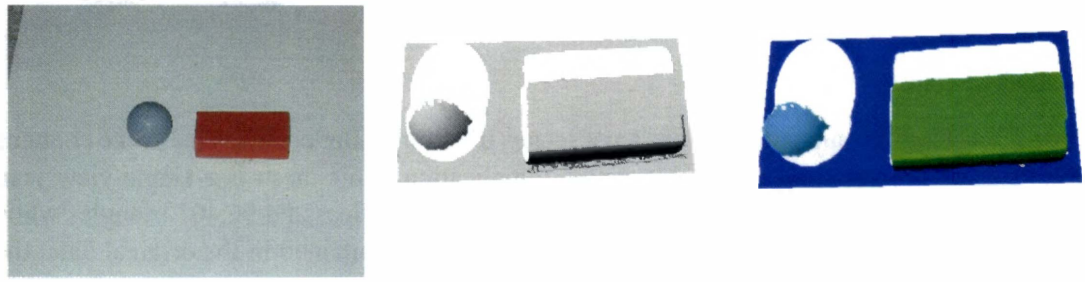


Figure 7.11: Part decomposition results for a y-shaped pipe fitting. The size of the pipe fitting is $2in \times 2in \times 2in$. This mesh is a single-view scan from the IVP Ranger System. The mesh consists of 16,254 vertices and 31,587 triangles while the decomposition consists of five parts. To reduce the noise contained in the original data, the mesh is smoothed for 30 iterations using the Laplacian mesh smoothing function included in the RapidForm software. Both original and smoothed mesh are pseudo-colored to illustrate the range depth variations on the surfaces. Red corresponds to smaller distances between the sensor and the object surface while blue corresponds to larger distances. The smoothed data is then segmented. (a) Photograph of original object, (b) rendering of original mesh, (c) pseudo-colored data for the original mesh, (d) rendering of smoothed mesh, (e) pseudo-colored data for the smoothed mesh, and (f) decomposed parts labeled in different colors.



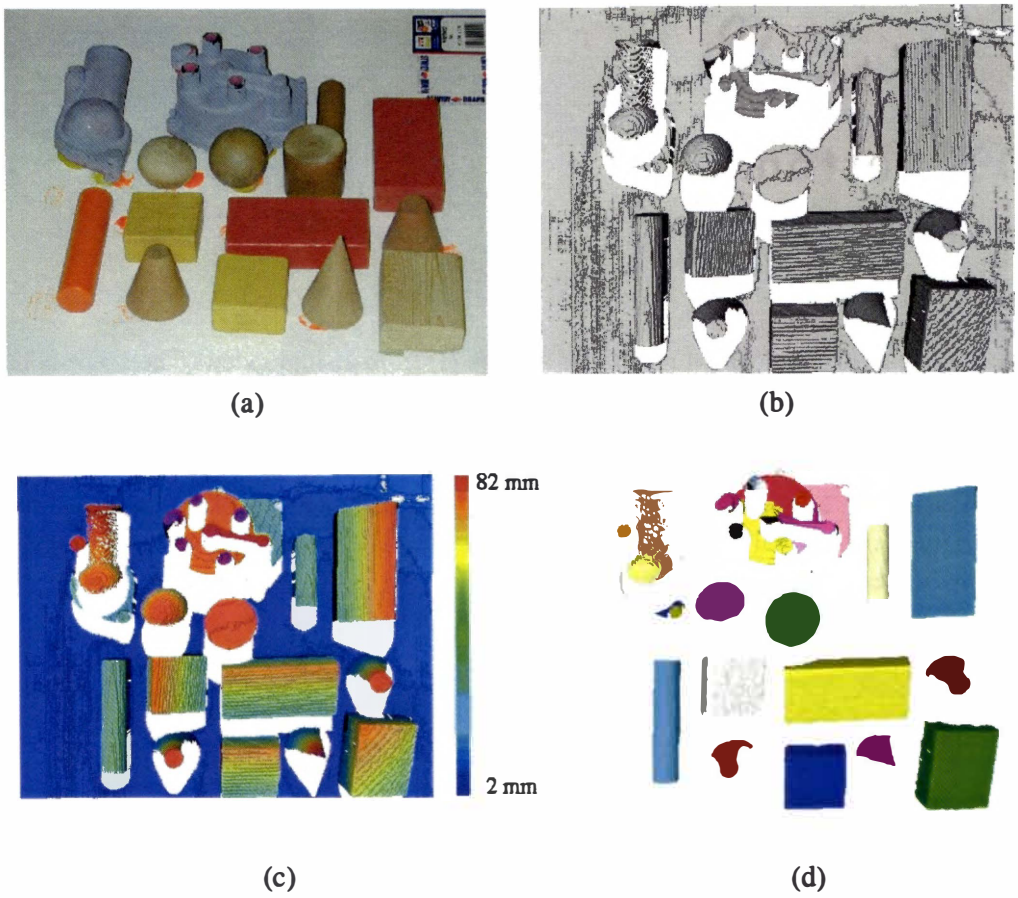
(a) (b) (c)

Figure 7.12: Part decomposition results for a scene (I) from a single view. The scene contains two blocks. This mesh is a single-view scan from the IVP Ranger System. The mesh consists of 20,156 vertices and 39,157 triangles while the decomposition consists of three parts including the background. To reduce the noise contained in the original data, the mesh is smoothed for 30 iterations using the Laplacian mesh smoothing function included in the RapidForm software. (a) Photograph of original object, (b) rendering of original mesh, and (c) decomposed parts labeled in different colors.



(a) (b) (c)

Figure 7.13: Part decomposition results for a scene (II) from a single view. This scene contains a block and a sphere. This mesh is a single-view scan from the IVP Ranger System. The mesh consists of 27,615 vertices and 53,266 triangles while the decomposition consists of three parts including the background. To reduce the noise contained in the original data, the mesh is smoothed for 30 iterations using the Laplacian mesh smoothing function included in the RapidForm software. (a) Photograph of original scene, (b) rendering of original mesh, and (c) decomposed parts labeled in different colors.



(a) (b)

(c) (d)

Figure 7.14: Part decomposition results for a scene (III) from a single view. This scene contains miscellaneous objects simulating a bin picking application. This mesh is a single-view scan from the IVP Ranger System. The mesh consists of 51,121 vertices and 96,401 triangles while the decomposition consists of 30 parts. To reduce the noise contained in the original data, the mesh is smoothed for 30 iterations using the Laplacian mesh smoothing function included in the RapidForm software. Since this scan is from a single view, the mesh leaves many objects as occluded and/or isolated and thus unconnected to other objects. The original mesh is pseudo-colored to illustrate the range depth variations on the surfaces. Red corresponds to smaller distances between the sensor and the object surface while blue corresponds to larger distances. (a) Photograph of original scene, (b) rendering of original mesh, (c) pseudo-colored data for the original mesh, and (d) decomposed parts labeled in different colors.

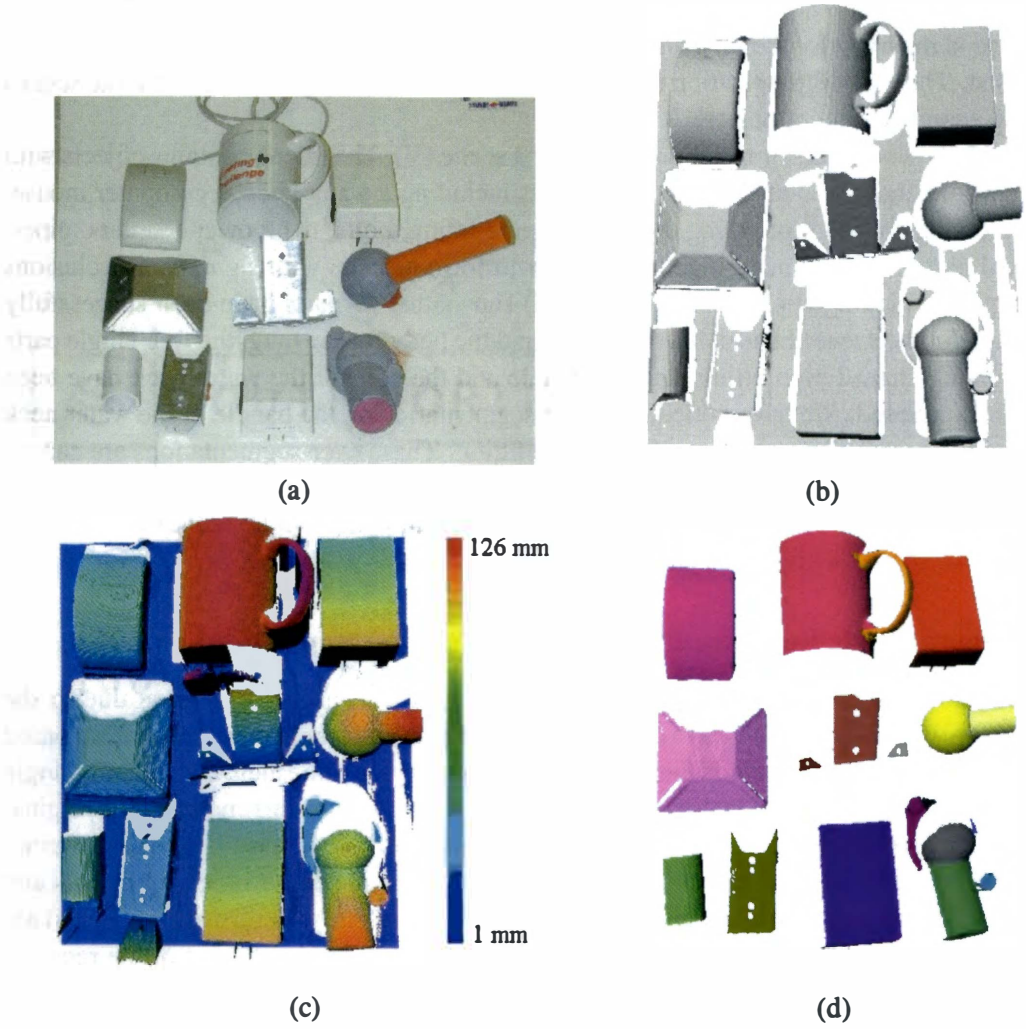


Figure 7.15: Part decomposition results for a scene (IV) from a single view. This scene contains miscellaneous objects simulating a bin picking application. This mesh is a single-view scan from the IVP Ranger System. The mesh consists of 90,237 vertices and 175,298 triangles while the decomposition consists of 18 parts. To reduce the noise contained in the original data, the mesh is smoothed for 30 iterations using the Laplacian mesh smoothing function included in the RapidForm software. Since this scan is from a single view, the mesh leaves many objects as occluded and/or isolated and thus unconnected to other objects. The original mesh is pseudo-colored to illustrate the range depth variations on the surfaces. Red corresponds to smaller distances between the sensor and the object surface while blue corresponds to larger distances. (a) Photograph of original scene, (b) rendering of original mesh, (c) pseudo-colored data for the original mesh, and (d) decomposed parts labeled in different colors.

this scene contains real objects including a cup, a computer mouse, two power adapters, a water neck, a bolt, and three mechanical parts. This scene better simulates a bin picking application. Fig. 7.15(d) shows that 18 parts have been successfully decomposed for the scene. Especially, the handle and the body for the mug and the handle and the ball for the water neck are correctly decomposed. The part decomposition results demonstrate that our algorithm can handle scenes containing a large variety of objects.

Fig. 7.16 shows part decomposition results for a scene (V). This scene contains objects with more occlusions than previous scenes. The objects including a water neck, a computer mouse, a cup, a crossing-shaped pipe fitting, a y-shaped pipe fitting, bolts, two power adapters, pipes, mechanical parts, etc. are piled together in order to form a scene with significant occlusions simulating a bin picking application. Fig. 7.16(d) shows that 49 parts have been successfully decomposed for the scene. Especially, the handle and the body for the mug, the four single parts for the crossing-shaped pipe fitting, and the handle and the ball for the water neck have been correctly decomposed. We have noticed the over-segmentation on the handle of the water neck and around the center of the crossing-shaped pipe fitting. These over-segmentations are caused by the resolution of the original mesh and have been reduced to the minima in the experiments. The part decomposition results demonstrate that our algorithm can handle scenes containing a large variety of shapes with significant occlusions.

7.1.2 Superquadric Representation Utilizing Part Decomposition

Superquadric representation of multi-part objects has been an unsolved problem due to the complex geometry involved. We have developed a superquadric representation algorithm based on part decomposition. Our algorithm first decomposes objects into their constituent, single parts, and then fits each individual part to a superquadric model. In this manner, the original superquadric representation problem is simplified into multiple, simpler problems of data fitting. This section demonstrates the capabilities of this strategy through a large variety of objects and scenes. Regular, globally deformed, and superquadrics with free-form deformations (FFDs) are explored. In addition, the performance of our algorithm is compared with that of the recover-and-select (RAS) algorithm (Leonardis et al., 1997) through experiments, which is the state of the art algorithm for superquadric representation. Multi-part objects and scenes containing multiple objects are explored separately.

Superquadric representation of objects and scenes are essential to applications such as object recognition, data transmission, scene visualization, etc. since superquadrics provide the most compact and parametric description about the objects and the scenes. To demonstrate the application of our superquadric representation strategy to automotive-related areas, we begin with automotive parts including a disc brake, a distributor cap, and a water neck. Next, single-view scans of a hammer, a dumbbell, a crossing-shaped pipe fitting, and a y-shaped pipe fitting are explored, which shows that superquadrics can be recovered from single-view 3D meshes of common tools and industrial parts. Finally, a set of scenes consisting of multiple objects with various geometries and occlusions are investigated to simulate waste area clean-up and bin picking applications. The data we have explored in the experiments can be classified into the following categories.

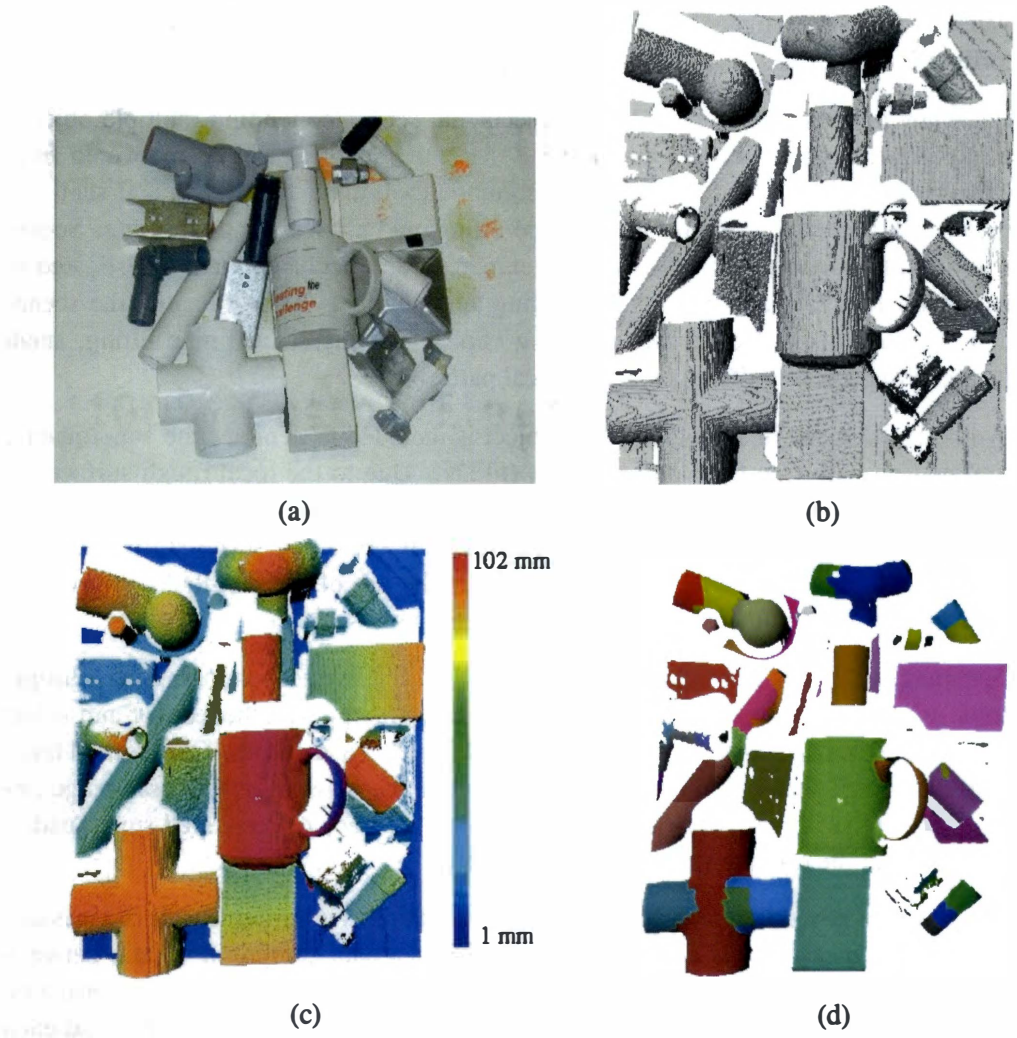


Figure 7.16: Part decomposition results for a scene (V) from a single view. This scene contains miscellaneous objects simulating a bin picking application. This mesh is a single-view scan from the IVP Ranger System. The mesh consists of 89,561 vertices and 169,479 triangles while the decomposition consists of 49 parts. To reduce the noise contained in the original data, the mesh is smoothed for 30 iterations using the Laplacian mesh smoothing function included in the RapidForm software. Since this scan is from a single view, the mesh leaves many objects as occluded and/or isolated and thus unconnected to other objects. The original mesh is pseudo-colored to illustrate the range depth variations on the surfaces. Red corresponds to smaller distances between the sensor and the object surface while blue corresponds to larger distances. (a) Photograph of original scene, (b) rendering of original mesh, (c) pseudo-colored data for the original mesh, and (d) decomposed parts labeled in different colors.

- Automotive parts simulating applications in reverse engineering, CAD, etc. involving automotive objects. Reconstructed 3D meshes for a disc brake, a distributor cap, and a water neck are investigated to demonstrate the application of our part decomposition algorithm in these areas.

- Single-view range scans of a variety of objects and scenes containing multiple objects. The experiments on these data demonstrate that superquadrics can be recovered from single-view 3D meshes. We have included common tools and industrial objects such as a hammer, a dumbbell, a crossing-shaped pipe fitting, and a y-shaped pipe fitting. Scenes containing multiple objects with various geometries and occlusions are investigated to simulate waste area clean-up and bin picking applications. To be specific, the scenes contain a water neck, a computer mouse, a cup, a crossing-shaped pipe fitting, small bolts, power adapters, PVC pipes, mechanical parts, etc.

- A synthetic 3D mesh of a football. This object is included to demonstrate superquadric representation with free-form deformations (FFDs). Due to the local rough surfaces of the football, regular and globally deformed superquadrics cannot accurately represent the geometry details of the football. Superquadrics with FFDs can successfully represent such small ridges. One of the small cylinders on the top of the distributor cap is also represented by a superquadric with FFDs due to its rough surface.

- Range images of a cup, a crossing-shaped pipe fitting, and two scenes containing multiple objects. We compare our superquadric representation strategy with the recover-and-select (RAS) algorithm (Leonardis et al., 1997) on these objects and scenes. Our method takes single-view range scans of 3D meshes as input while the RAS algorithm uses range images. The comparison is performed in terms of the accuracy of recovered superquadric parameters and the running time of the algorithms.

Since an essential advantage of superquadric representation is compactness, it is necessary to define a metric to measure this compactness. We define a data compression ratio between superquadric and triangle mesh representation. For a triangle mesh representation consisting of a set of vertices and triangles, each vertex is represented by (x, y, z) coordinates, and each triangle is represented by indices of its three composite vertices. Assuming that 32 bits in the memory of a computer are used to represent integer- and float-typed variables, the data compression ratio is defined as

$$DCR = \frac{3 \times N(V) + 3 \times N(T)}{m \times N(SQ)}, \tag{7.1}$$

where $N(V)$ denotes the number of vertices in a mesh representation and $N(T)$ denotes the number of triangles. The variable $N(SQ)$ represents the number of superquadrics recovered for an object. Each superduadric is defined by m parameters, where the constant m is 11 for regular superquadrics or 15 for superquadrics with deformations.

Similar to Table 7.1, Table 7.2 provides a list of objects and scenes used in the experiments of superquadric representation. This table briefly describes the objects and scenes regarding the number of vertices, the number of triangles, the number of recovered superquadrics, and data

Table 7.2: List of objects and scenes for superquadric representation.

Object Name	Figure Number	Number of Vertices	Number of Triangles	Number of SQs	Data Compression Ratio (DCR)
Disc brake	7.17	37,171	73,394	2	11056.5
Distributor cap	7.18	58,975	117,036	13	2707.86
Water neck	7.19	58,784	117,564	9	3918.84
Bore pin	7.20	37,450	74,896	4	5617.3
Hammer	7.21	18,767	36,478	2	5524.5
Dumbbell	7.22	11,610	22,542	3	2276.8
"Cross" pipe fitting	7.23	17,963	35,238	4	2660.05
"Y" pipe fitting	7.24	16,254	31,587	4	1913.64
scene (I)	7.25	20,156	39,157	3	5931.3
scene (II)	7.26	27,615	53,266	3	8088.1
scene (III)	7.27	51,121	96,401	30	983.48
scene (IV)	7.28	90,237	175,298	18	2950.39
scene (V)	7.29	89,561	169,479	49	1057.31

(a) (b) (c)

Figure 7.17: Superquadric representation results for a disc brake. This mesh is a reconstruction from multiple range images using the IVP Ranger System. The mesh consists of 37,171 vertices and 73,394 triangles while the decomposition consists of two parts. Each decomposed part is fitted to a superquadric model. Each superquadric is rendered in 3D using quad meshes based on recovered superquadric parameters. (a) Rendering of original mesh, (b) decomposed parts labeled in different colors, and (c) rendering of recovered superquadrics.

compression ratio of superquadric representation. We have two major sources for each mesh. We have either generated the meshes using our range scanners or used meshes from other laboratories or commercial companies. For meshes obtained in our laboratory, we have used either reconstructed surfaces or single-view scans of the objects. As shown in Table 7.2, the mesh for the bore pin is a reconstruction from the U. S. Army TACOM National Automotive Center. The remaining meshes are generated in our laboratory from the IVP Ranger System (Integrated Vision Products, 2000). Among them, the meshes for the disc brake, the distributor cap, and the water neck are reconstructions from multi-view scans (Sun et al., 2002). The meshes for the hammer, the dumbbell, the crossing-shaped pipe fitting, the y-shaped pipe fitting, and the five scenes containing miscellaneous objects are from single-view scans. By conducting experiments on both reconstructed and single-view meshes, we demonstrate the versatility of our superquadric representation algorithm. Multi-part objects and multi-object scenes are explored separately in the following.

Multi-Part Objects

The following eight figures demonstrate superquadric representation results for a variety of multi-part objects. For the disc brake, the distributor cap, the water neck shown in the following three figures, the meshes are reconstructions from multi-view range images. The range images are acquired in our laboratory from the IVP Ranger System.

Fig. 7.17 shows superquadric representation results for the disc brake. Using our superquadric representation strategy, the disc brake is first decomposed into single parts. Each decomposed part is next fitted to a superquadric model. Fig. 7.17 shows two decomposed parts for the disc brake. The recovered and rendered superquadrics shown in Fig. 7.17(c) demonstrate that the

(a) (b) (c)

Figure 7.18: Superquadric representation results for a distributor cap. This mesh is a reconstruction from multiple range images using the IVP Ranger System. The mesh consists of 58,975 vertices and 117,036 triangles while the decomposition consists of 13 parts. Each decomposed part is fitted to a superquadric model. Each superquadric is rendered in 3D using quad meshes based on recovered superquadric parameters. (a) Rendering of original mesh, (b) decomposed parts labeled in different colors, and (c) rendering of recovered superquadrics.

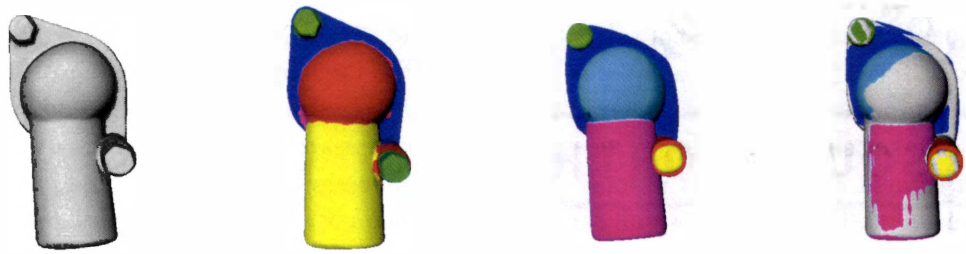
disc brake is correctly represented by two superquadrics in terms of shape, size, orientation, position, etc. However, the hole at the center of the disc brake fails to be represented since superquadrics can only represent solids (genus equal to zero). Despite the hole, superquadrics recovered for the disc brake approximately capture the fundamental information of the disc brake. As to the compactness achieved by superquadric representation, the original mesh for the disc brake consists of 37,171 vertices and 73,394 triangles while the recovered superquadric consist of only 30 parameters in total. The data compression ratio DCR for the disc brake is evaluated as 11056.5. The significantly lower storage cost of the superquadric representation enables tasks such as visualization, transmission, manipulation, CAD modeling, etc. to be performed in real time, which is a huge advantage over the mesh representation.

Fig. 7.18 shows superquadric representation results for the distributor cap. Fig. 7.18(b) illustrates that almost every single part has been decomposed from the original object. Superquadrics recovered for the decomposed parts are rendered in Fig. 7.18(c). We observe that recovered superquadrics are able to correctly represent the size, shape, orientation, and position information of the distributor cap with a low storage and manipulation cost. To be specific, the original mesh for the distributor cap consists of 58,975 vertices and 117,036 triangles while the recovered superquadrics consists of only 195 parameters in total. The data compression ratio DCR for the distributor cap is evaluated as 2707.86. Table 7.3 shows ground truth values and recovered superquadric parameters for one of the cylinders on the top of the distributor cap. The recovered parameters are accurate compared with the ground truth values. This demonstrates the accuracy of the superquadric representation from our algorithm.

Fig. 7.19 shows superquadric representation results for the water neck. Fig. 7.19(b) illustrates that every single part has been decomposed for the water neck. Superquadrics recovered

Table 7.3: Recovered superquadric parameters for a decomposed part of a distributor cap. The part is one of the small cylinders on the top of the distributor cap.

	a_1	a_2	a_3	ε_1	ε_2
Ground truth	15.2	15.6	20.1	0.1	1.0
Recovered parameters	16.45	15.67	20.42	0.12	0.96



(a) (b) (c) (d)

Figure 7.19: Superquadric representation results for a water neck. This mesh is a reconstruction from multiple range images using the IVP Ranger System. The mesh consists of 58,784 vertices and 117,564 triangles while the decomposition consists of nine parts. Each decomposed part is fitted to a superquadric model. Each superquadric is rendered in 3D using quad meshes based on recovered superquadric parameters. Superquadrics and original mesh for the water neck are rendered in the same coordinate system to illustrate the fitting error. (a) Rendering of original mesh, (b) decomposed parts labeled in different colors, (c) rendering of recovered superquadrics, and (d) merged rendering of original mesh and recovered superquadrics in the same coordinate system.

Table 7.4: Recovered superquadric parameters for three decomposed parts of a water neck. The unit is mm.

Object	Parameter	a_1	a_2	a_3	ε_1	ε_2
Handle	Ground truth	39.7	39.4	67.6	0.1	1.0
	Recovered parameters	40.23	40.58	66.83	0.13	0.98
Ball	Ground truth	50.0	47.6	56.0	1.0	1.0
	Recovered parameters	51.62	47.56	54.28	1.02	0.95
Cylinder	Ground truth	16.5	17.8	44.2	0.1	1.0
	Recovered parameters	17.56	17.94	43.38	0.11	0.95

for the decomposed parts are rendered in Fig. 7.19(c). In Fig. 7.19(d), superquadrics and original mesh for the water neck are rendered in the same coordinate system to illustrate the fitting error between the raw data and the recovered superquadrics. This figure shows that recovered superquadrics correctly fit the original data with small fitting errors. In addition, we observe that recovered superquadrics are able to correctly represent the size, shape, orientation, and position information of the water neck with a significantly low storage and manipulation cost. To be specific, the original mesh for the distributor cap consists of 58,784 vertices and 117,564 triangles while the recovered superquadrics consists of only 135 parameters in total. The data compression ratio DCR for the distributor cap is evaluated as 3918.84.

Table 7.4 shows ground truth values and recovered superquadric parameters for three decomposed parts for the water neck including the handle, the ball, and a small cylinder next to the handle. We observe that the recovered superquadric parameters are accurate compared with the ground truth values for these three parts. This demonstrates the accuracy of the recovered superquadrics and the capabilities of our algorithm.

Fig. 7.20 show superquadric representation results for a small bore pin. The mesh for the bore pin is a reconstruction from the U. S. Army TACOM National Automotive Center. Fig. 7.20(b) shows four decomposed parts for the bore pin. Each of the decomposed part is fitted to a superquadric model. Fig. 7.20(c) shows four recovered superquadrics. Each superquadric is rendered in 3D using quad meshes based on recovered superquadric parameters. We can hardly observe any visual difference between the superquadric and mesh representation for the bore pin while the superquadric representation has a significantly lower storage cost compared with triangle meshes. To be specific, the mesh of the bore pin consists of 37,450 vertices and 74,896 triangles while the superquadric representation only consists of 60 parameters. The data compression ratio DCR is 5617.3. This demonstrates an essential advantage of superquadrics as compact but powerful primitives over triangle meshes.

(a) (b) (c)

Figure 7.20: Superquadric representation results for a small bore pin. This mesh is a reconstruction from a point cloud data set generated at the U. S. Army TACOM National Automotive Center. The mesh consists of 37,450 vertices and 74,896 triangles while the decomposition consists of four parts. Each of the decomposed part is fitted to a superquadric model. Each superquadric is rendered in 3D using quad meshes based on recovered superquadric parameters. (a) Rendering of original mesh, (b) decomposed parts labeled in different colors, and (c) rendering of recovered superquadrics.

(a) (b) (c)

Figure 7.21: Superquadric representation results for a hammer. This mesh is a single-view scan from the IVP Ranger System. The mesh consists of 18,767 vertices and 36,478 triangles while the decomposition consists of two parts. Each decomposed part is fitted to a superquadric model. Superquadrics are then rendered in 3D using quad meshes based on recovered superquadric parameters. (a) Rendering of original mesh, (b) decomposed parts labeled in different colors, and (c) rendering of recovered superquadrics.

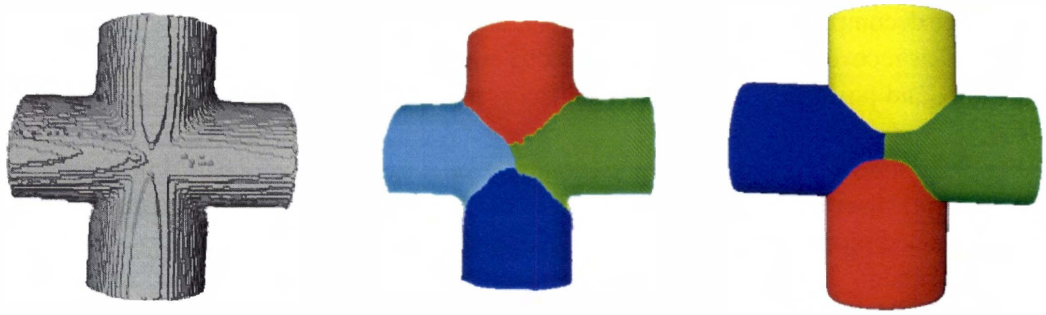
Figs. 7.21 and 7.22 show superquadric representation results for a hammer and a dumbbell. The meshes for both objects are single-view scans in our laboratory from the IVP Ranger System. For the hammer, Fig. 7.21(b) shows two decomposed parts, and Fig. 7.21(c) shows two recovered superquadrics. We observe that recovered superquadrics represent the hammer well. The data compression ratio achieved for the hammer is 5524.5. For the dumbbell, Fig. 7.22(b) shows three decomposed parts, and Fig. 7.22(c) shows three recovered superquadrics. It can be observed that recovered superquadrics correctly represent the dumbbell regarding size, shape, orientation, and position information. The data compression ratio achieved for the dumbbell is 2276.8.

Fig. 7.23 shows superquadric representation results for a crossing-shaped pipe fitting. The mesh for this object is a single-view scan in our laboratory from the IVP Ranger System. The crossing-shaped pipe fitting is decomposed into four cylinders. Each cylinder is fitted to a superquadric model. Fig. 7.23(c) shows that each decomposed part is correctly represented by a superquadric in terms of shape, size, orientation, position, etc. Moreover, the original mesh for the crossing-shaped pipe fitting consists of 17,963 vertices and 35,238 triangles while the recovered superquadrics consist of 60 parameters in total. The data compression ratio DCR is evaluated as 2660.05. The significantly lower storage costs of the superquadric representation for the crossing-shaped pipe enables tasks such as visualization, transmission, manipulation, CAD modeling, etc. to be performed in real time, which is a huge advance to the mesh representation.

Similarly, Fig. 7.24 shows superquadric representation results for a y-shaped pipe fitting. The mesh for this object is a single-view scan in our laboratory from the IVP Ranger System. As shown in Fig. 7.24(b), our part decomposition algorithm is able to decompose the pipe fitting

(a) (b) (c)

Figure 7.22: Superquadric representation results for a dumbbell. This mesh is a single-view scan from the IVP Ranger System. The mesh consists of 11,610 vertices and 22,542 triangles while the decomposition consists of three parts. Each decomposed part is fitted to a superquadric model. Superquadrics are then rendered in 3D using quad meshes based on recovered superquadric parameters. (a) Rendering of original mesh, (b) decomposed parts labeled in different colors, and (c) rendering of recovered superquadrics.



(a) (b) (c)

Figure 7.23: Superquadric representation results for a crossing-shaped pipe fitting. This mesh is a single-view scan from the IVP Ranger System. The mesh consists of 17,963 vertices and 35,238 triangles while the decomposition consists of four parts. Each decomposed part is fitted to a superquadric model. Superquadrics are then rendered in 3D using quad meshes based on recovered superquadric parameters. (a) Rendering of original mesh, (b) decomposed parts labeled in different colors, and (c) rendering of recovered superquadrics.

(a) (b) (c)

Figure 7.24: Superquadric representation results for a y-shaped pipe fitting. This mesh is a single-view scan from the IVP Ranger System. The mesh consists of 16,254 vertices and 31,587 triangles while the decomposition consists of five parts. Each decomposed part is fitted to a superquadric model. Superquadrics are then rendered in 3D using quad meshes based on recovered superquadric parameters. (a) Rendering of original mesh, (b) decomposed parts labeled in different colors, and (c) rendering of recovered superquadrics.

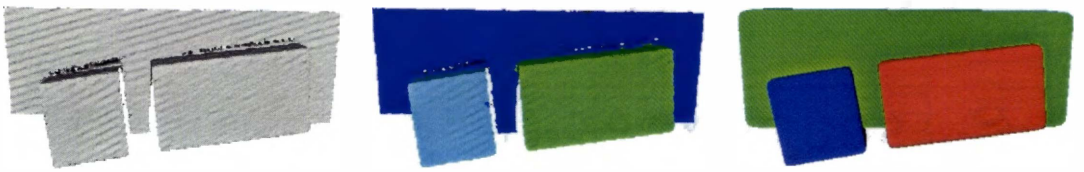
into five single parts as shown in Fig. 7.24(b), which is consistent with human perception. Each of the five decomposed parts is fitted to a superquadric model. Fig. 7.24(c) illustrates that each decomposed part is correctly represented by a superquadric in terms of shape, size, orientation, position, etc. Moreover, the original mesh for the y-shaped pipe fitting consists of 16,254 vertices and 31,587 triangles while the recovered superquadrics consist of totally 75 parameters. The data compression ratio DCR is evaluated as 1913.64.

Scenes Containing Multiple Objects

The following five figures demonstrate superquadric representation results for a variety of scenes containing multiple objects. The range images are acquired and triangulated in our laboratory from the IVP Ranger System (Integrated Vision Products, 2000). Each mesh is a single-view scan.

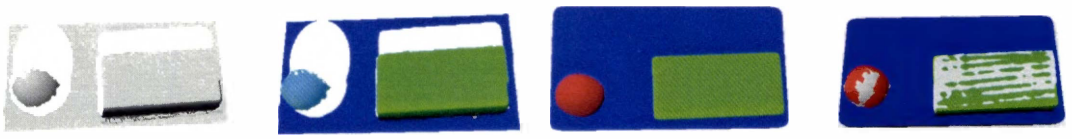
Fig. 7.25 shows superquadric representation results for a scene (I). This scene contains two separate small blocks. Fig. 7.25(b) shows the three decomposed parts including the two blocks and the background. Fig. 7.25(c) illustrates that superquadrics have been successfully recovered for the two decomposed parts in the scene. The recovered superquadrics reflect the correct size, position, and orientation information for the original objects in the scene. The data compression ratio DCR between the superquadric representation and the mesh achieved for this scene is evaluated as 5931.3.

Fig. 7.26 shows superquadric representation results for a scene (II). This scene contains two isolated objects including a block (planar surfaces) and a sphere (cylindrical surfaces). Fig. 7.26(b) shows three decomposed parts including the block, the sphere, and the background. Fig. 7.26(c) shows that superquadrics have been successfully recovered for the two decomposed



(a) (b) (c)

Figure 7.25: Superquadric representation results for a scene (I) from a single view. This scene contains two blocks. This mesh is a single-view scan from the IVP Ranger System. The mesh consists of 20,156 vertices and 39,157 triangles while the decomposition consists of three parts including the background. Since this scan is from a single view, the mesh leaves the objects as isolated and thus unconnected to each other. Each decomposed part is fitted to a superquadric model. Superquadrics are then rendered in 3D using quad meshes based on recovered superquadric parameters. (a) Rendering of original mesh, (b) decomposed parts labeled in different colors, and (c) rendering of recovered superquadrics.



(a) (b) (c) (d)

Figure 7.26: Superquadric representation results for a scene (II) from a single view. This scene contains a block and a sphere. This mesh is a single-view scan from the IVP Ranger System. The mesh consists of 27,615 vertices and 53,266 triangles while the decomposition consists of three parts including the background. Since this scan is from a single view, the mesh leaves the objects as isolated and thus unconnected to each other. Each decomposed part is fitted to a superquadric model. Superquadrics are then rendered in 3D using quad meshes based on recovered superquadric parameters. Superquadrics and original mesh for the scene are rendered in the same coordinate system to illustrate the fitting error. (a) Rendering of original mesh, (b) decomposed parts labeled in different colors, (c) rendering of recovered superquadrics, and (d) merged rendering of original scene and recovered superquadrics in the same coordinate system

objects in the scene. In Fig. 7.26(d), superquadrics and original mesh for the scene are rendered in the same coordinate system to illustrate the fitting error between the raw data and the recovered superquadrics. This figure shows that recovered superquadrics correctly fit the original data with small fitting errors. In addition, the recovered superquadrics reflect the correct size, position, and orientation information for the original objects in the scene. The data compression ratio DCR between the superquadric representation and the mesh for this scene is evaluated as 8088.1.

Fig. 7.27 shows superquadric representation results for a scene (III). This scene contains a larger variety of shapes. The scene contains miscellaneous objects including five blocks, two spheres, three cylinders, three cones, a water neck, and a distributor cap simulating a bin picking application. Some objects in this scene are separate and some are linked to each other. Since this scan is from a single view, the mesh leaves many objects as occluded and/or isolated and thus unconnected to other objects. Fig. 7.27(b) shows that this scene has been successfully decomposed into 30 single parts. Fig. 7.27(b) shows recovered superquadrics for each decomposed part in the scene. The data compression ratio DCR between the superquadric representation and the mesh for this scene is evaluated as 983.48.

Fig. 7.28 shows superquadric representation results for a scene (IV). This scene contains real objects including a cup, a computer mouse, two power adapters, a water neck, a bolt, and three mechanical parts simulating a bin picking application. Fig. 7.28(b) shows that 18 single parts have been successfully decomposed. Fig. 7.28(b) shows recovered superquadrics for each decomposed part in the scene. The data compression ratio DCR between the superquadric representation and the mesh for this scene is evaluated as 2950.39.

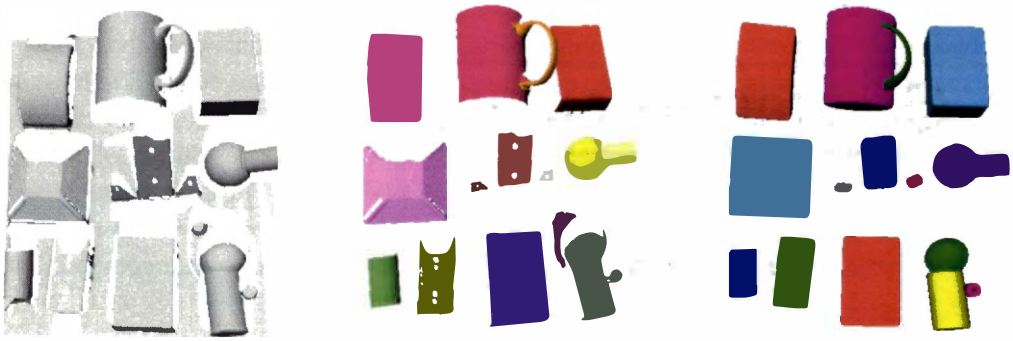
Fig. 7.29 shows superquadric representation results for a scene (V). This scene contains a water neck, a computer mouse, a cup, a crossing-shaped pipe fitting, a y-shaped pipe fitting, bolts, two power adapters, pipes, mechanical parts, etc. with significant occlusions. Fig. 7.29(b) shows that 49 single parts have been successfully decomposed for this scene. We have noticed the over-segmentation on the handle of the water neck, and around the center of the crossing-shaped pipe fitting. These over-segmentations are caused by the resolution of the original mesh and have been reduced to the minima in the experiments. For each decomposed part, a superquadric model is recovered and shown in Fig. 7.29(c). We observe that recovered superquadrics have correctly represented the original scene with a huge data compression ratio DCR of 1057.31.

Superquadric Representation with FFDs

Superquadrics with global deformations can only capture global characteristics of objects. Free-form deformations (FFDs) are necessary when local details of the objects need to be represented as well. We implement the superquadric representation algorithm with FFDs described in Section 3.4. Fig. 7.30 shows superquadric representation results with FFDs for a football. The football contains local details on the surface as shown in Fig. 7.30(a). Neither a regular nor a globally deformed superquadric can represent these details accurately. We observe that the initial superquadric model without FFDs shown in Fig. 7.30(b) can only represent the global shape of the football, and fails to represent the details on the surface. Therefore, FFDs are explored to solve this problem. Fig. 7.30(c) shows the generated control points surrounding the initial

(a) (b) (c)

Figure 7.27: Superquadric representation results for a scene (III) from a single view. This scene contains miscellaneous objects simulating a bin picking application. This mesh is a single-view scan from the IVP Ranger System. The mesh consists of 51,121 vertices and 96,401 triangles while the decomposition consists of 30 parts. Since this scan is from a single view, the mesh leaves the objects as isolated and thus unconnected to each other. Each decomposed part is fitted to a superquadric model. Superquadrics are then rendered in 3D using quad meshes based on recovered superquadric parameters. (a) Rendering of original mesh, (b) decomposed parts labeled in different colors, and (c) rendering of recovered superquadrics.



(a) (b) (c)

Figure 7.28: Superquadric representation results for a scene (IV) from a single view. This scene contains miscellaneous objects simulating a bin picking application. This mesh is a single-view scan from the IVP Ranger System. The mesh consists of 90,237 vertices and 175,298 triangles while the decomposition consists of 18 parts. Since this scan is from a single view, the mesh leaves the objects as isolated and thus unconnected to each other. Each decomposed part is fitted to a superquadric model. Superquadrics are then rendered in 3D using quad meshes based on recovered superquadric parameters. (a) Rendering of original mesh, (b) decomposed parts labeled in different colors, and (c) rendering of recovered superquadrics.

(a) (b) (c)

Figure 7.29: Superquadric representation results for a scene (V) from a single view. This scene contains miscellaneous objects simulating a bin picking application. This mesh is a single-view scan from the IVP Ranger System. The mesh consists of 89,561 vertices and 169,479 triangles while the decomposition consists of 49 parts. Since this scan is from a single view, the mesh leaves the objects as isolated and thus unconnected to each other. Each decomposed part is fitted to a superquadric model. Superquadrics are then rendered in 3D using quad meshes based on recovered superquadric parameters. (a) Rendering of original mesh, (b) decomposed parts labeled in different colors, and (c) rendering of recovered superquadrics.

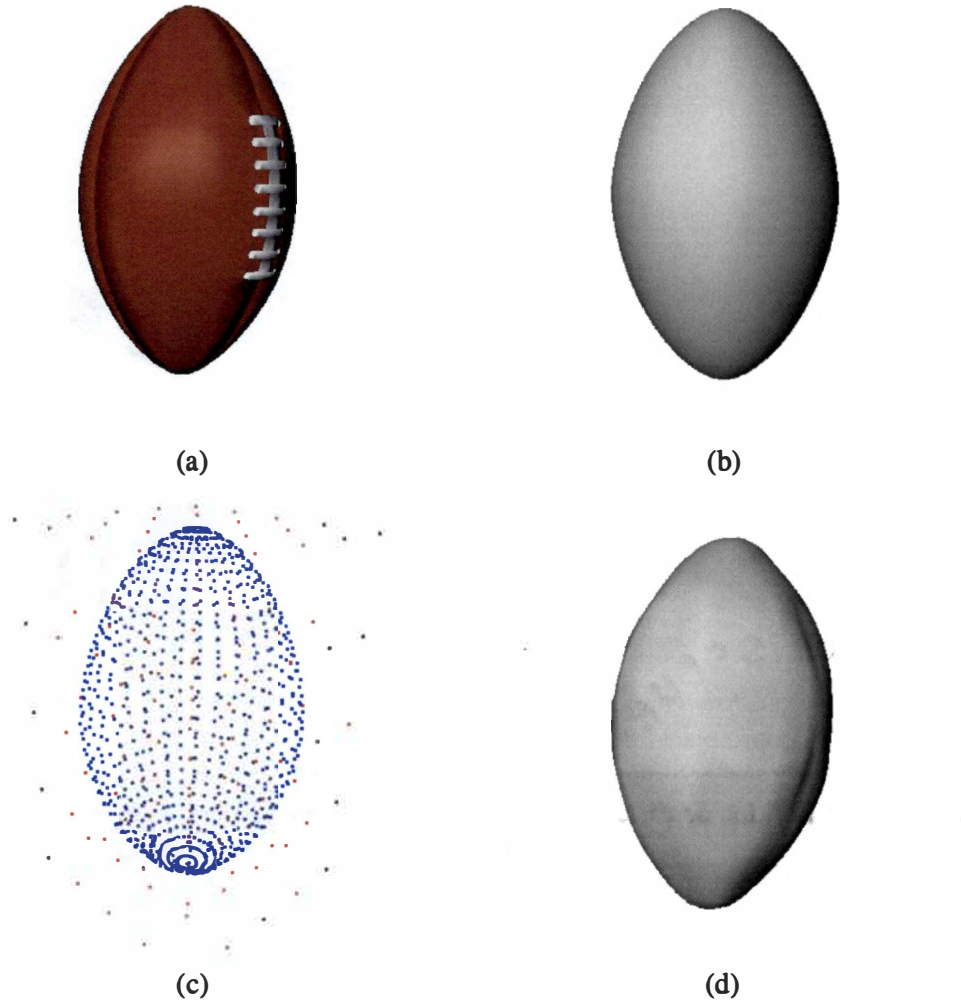


Figure 7.30: Superquadric representation results with FFDs for a football. This mesh is downloaded from the online VRML gallery (VRML Model Gallery, OCNUS Company, 2002). The mesh consists of 1,269 vertices and 2,222 triangles. A set of $5 \times 5 \times 5$ control points is generated on the bounding box of the football. It takes seven iterations to converge to the final model from the initial superquadric. (a) Photograph of original object, (b) rendering of the initial superquadric model without FFDs, (c) initial control points (red) and rendered superquadric points (blue), and (d) rendering of recovered superquadric model with FFDs.

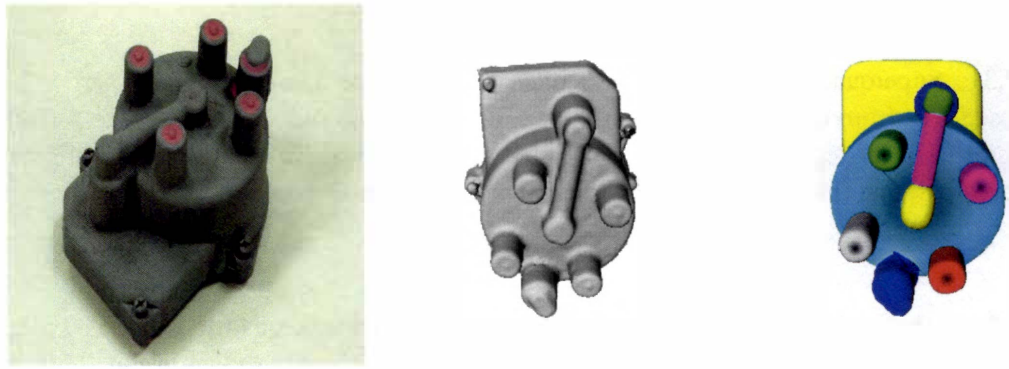
(a) (b)

Figure 7.31: Superquadric representation results with FFDs for one decomposed part. The decomposed part from the top of the distributor cap contains too many details on the surface to be represented by regular or globally deformed superquadrics. (a) Rendering of original decomposed part and (b) rendering of recovered superquadric model with FFDs.

superquadric. The initial superquadric model is transformed to a superquadric with FFDs after seven iterations. The superquadric with FFDs is rendered in 3D and is shown in Fig. 7.30(d). This figure illustrates that the superquadric with FFDs is able to represent local details on the surface of the football.

Fig. 7.31 shows superquadric representation results with FFDs for the two decomposed parts of the distributor cap including the small cylinder and the base. The decomposed part (one of the small cylinders on the top of the distributor cap) contain too many details on the surface to be represented by regular or globally deformed superquadrics. The small cylindrical object shown in Fig. 7.31(a) is not a smooth cylinder and has many local details on its surface. Fig. 7.31(b) shows a recovered superquadric with FFDs for the small cylindrical object. We observe that the final superquadric model with FFDs represents the surface details of the object well.

Fig. 7.32 shows superquadric representation results for the distributor cap incorporating FFDs. Combining the superquadrics with FFDs shown in Figs. 7.31(b), Fig. 7.32(c) shows final superquadric representation results for the distributor cap including one superquadric with FFDs. Fig. 7.32(d) illustrates that correct superquadrics have been recovered for every decomposed part of the distributor cap.



(a) (b) (c)

Figure 7.32: Superquadric representation results with FFDs for a distributor cap. Two decomposed parts including a cylinder on the top and the base of the distributor cap are represented by superquadrics with FFDs since they contain too much details on their surfaces to be represented by regular or globally deformed superquadrics. The remaining decomposed parts are represented by regular or globally deformed superquadrics. (a) Photograph of original object, (b) rendering of original mesh, and (c) rendering of recovered superquadrics with FFDs.

7.1.3 Comparison with the Recover-and-Select Algorithm

Among the existing algorithms for superquadric representation of multi-part objects, the recover-and-select algorithm (Leonardis et al., 1997) is the state of the art algorithm using range images. This algorithm is able to recover superquadrics from multi-part objects without pre-segmentation. In contrast, our proposed superquadric representation algorithm first applies part decomposition to triangle meshes. We compare our proposed superquadric representation algorithm with the recover-and-select algorithm through experiments.

Fig. 7.33 shows superquadric representation results for a mug using the recover-and-select and our proposed algorithms. The single-view range image is acquired and triangulated in our laboratory from the IVP Ranger Scanner System (Integrated Vision Products, 2000). The recover-and-select algorithm uses the single-view range image while our algorithm uses a mesh triangulated from the same range image. Fig. 7.33(c) illustrates that only one superquadric is recovered for the body of the mug while the handle fails to be represented using the recover-and-select algorithm. This is due to the fact that the recover-and-select algorithm is unable to handle global deformations in its implementation. Fig. 7.33(e) shows that both the body and the handle of the mug have been successfully decomposed. The recovered and rendered superquadrics shown in Fig. 7.33(f) demonstrate that our algorithm is able to successfully recover superquadrics for both parts of the mug, which also indicates that our algorithm is capable of handling globally deformed superquadrics. In addition, it takes approximately 6 minutes to recover superquadrics from the mug for the recover-and-select algorithm while our algorithm only needs approximately 1 minute in total for both part decomposition and superquadric fitting.

Fig. 7.34 shows superquadric representation results for a crossing-shaped pipe fitting using the recover-and-select and our proposed algorithms. Fig. 7.34(c) illustrates that three superquadrics are recovered for the pipe fitting using the recover-and-select algorithm. Fig. 7.34(e) shows that four single parts have been decomposed using our part decomposition algorithm. Correspondingly, four superquadrics have been recovered for the pipe fitting using our algorithm and are shown in Fig. 7.34(f).

The recovered superquadric parameters for the crossing-shaped pipe fitting using the two algorithms are tabulated in Table 7.5. We assume that this pipe fitting is composed of four penetrating cylinders of the same size and shape. Table 7.5 provides ground truth values for such a single cylinder. Comparing the recovered superquadric parameters with the ground truth values, we observe that the superquadrics recovered using our method are more accurate than those using the recover-and-select algorithm. In particular, all the three superquadrics recovered from the recover-and-select algorithm have inaccurate parameters for a_3 and ε_2. In contrast, all the superquadric parameters recovered from our algorithm are accurate except for a_1, which is reasonable since only single-view scans (partial surfaces) are used in the superquadric fitting. Table 7.5 demonstrates that our superquadric representation algorithm recovers much more accurate superquadrics than the recover-and-select algorithm. As to the computational time, it takes approximately 10 minutes for the recover-and-select algorithm to recover superquadrics from the crossing-shaped pipe fitting while our algorithm only needs approximately 42 seconds for part decomposition, and 30 seconds for superquadric fitting. In summary, our algorithm recovers more accurate superquadrics with a much lower computational cost.

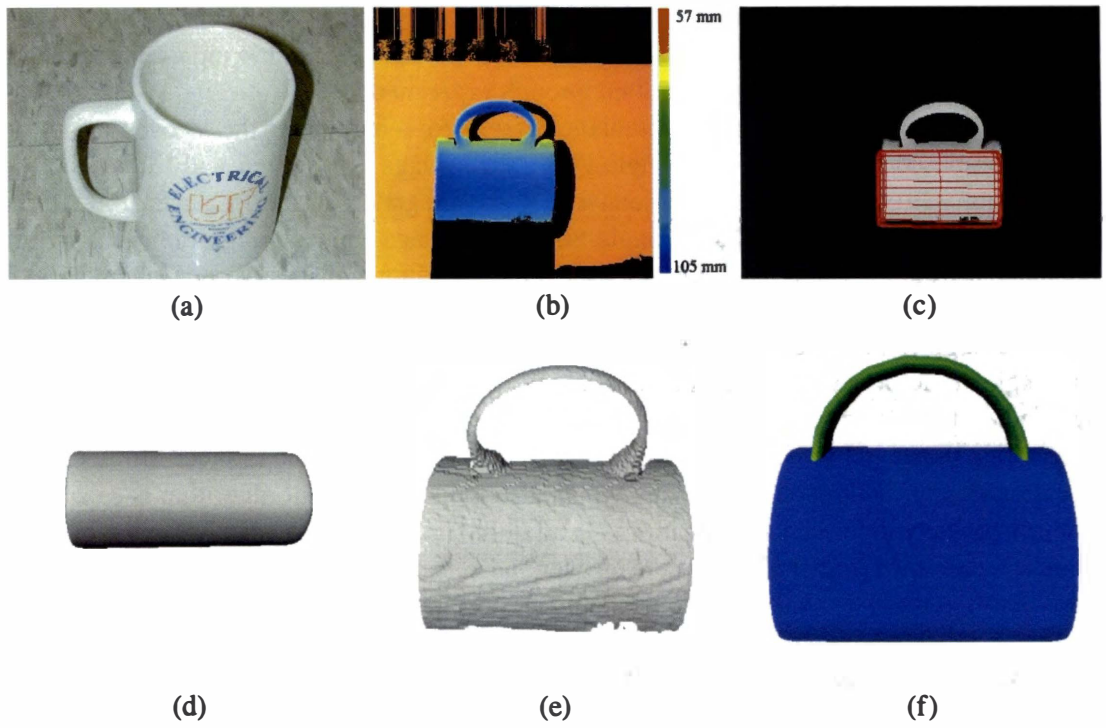


Figure 7.33: Comparison of superquadric representation results for a mug. Superquadrics are recovered for the object using two algorithms including our superquadric representation strategy utilizing part decomposition and the recover-and-select algorithm. The range data is a single-view scan from the IVP Ranger System. The original range image is pseudo-colored to illustrate the range depth variations on the surfaces. Red corresponds to larger distances between the sensor and the object surface while blue corresponds to smaller distances. For the recover-and-select algorithm, a single-view range image is used, and no pre-segmentation is involved. For our method, the mesh consists of 14,211 vertices and 27,547 triangles while the decomposition consists of two parts. Each of the decomposed parts is fitted to a superquadric model. (a) Photograph of original object, (b) pseudo-colored range image of original object, (c) recovered superquadrics using the recover-and-select algorithm and plotted by red wireframes, (d) rendering of superquadrics recovered from the recover-and-select algorithm, (e) rendering of original mesh, and (f) rendering of recovered superquadrics using our superquadric representation algorithm.

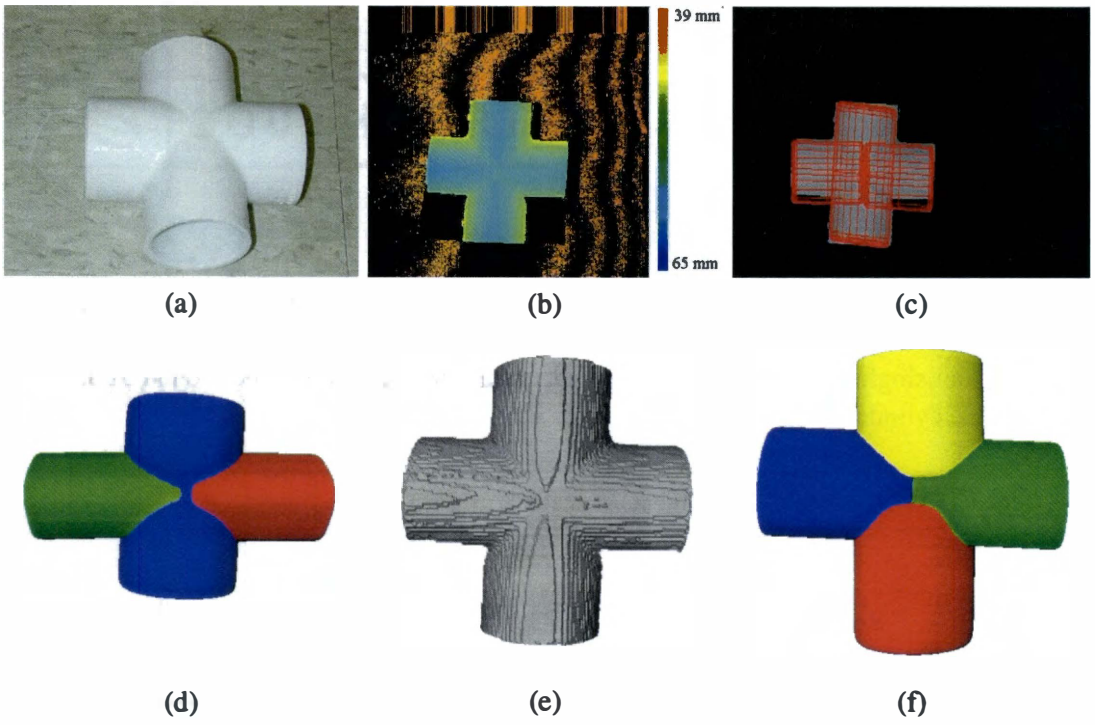


Figure 7.34: Comparison of superquadric representation results for a crossing-shaped pipe fitting. Superquadrics are recovered for the object using two algorithms including our superquadric representation strategy utilizing part decomposition and the recover-and-select algorithm. The range data is a single-view scan from the IVP Ranger System. The original range image is pseudo-colored to illustrate the range depth variations on the surfaces. Red corresponds to larger distances between the sensor and the object surface while blue corresponds to smaller distances. For the recover-and-select algorithm, a single-view range image is used, and no pre-segmentation is involved. For our method, the mesh consists of 17,963 vertices and 35,238 triangles while the decomposition consists of four parts. Each of the decomposed parts is fitted to a superquadric model. (a) Photograph of original object, (b) pseudo-colored range image of original object, (c) recovered superquadrics using the recover-and-select algorithm and plotted by red wireframes, (d) rendering of superquadrics recovered from the recover-and-select algorithm, (e) rendering of original mesh, and (f) rendering of recovered superquadrics using our superquadric representation algorithm.

Table 7.5: Recovered superquadric parameters for a crossing-shaped pipe fitting. Both recover-and-select and our proposed algorithms are used. The crossing-shaped pipe fitting is assumed to consist of four single penetrating cylinders of the same size and shape. The ground truth defines such a single cylinder.

Algorithm	Object	a_1	a_2	a_3	ε_1	ε_2
Ground truth		28.0	28.0	29.0	0.1	1.0
Recover-	Part I	42.83	42.16	65.68	0.1	0.93
And-	Part II	28.0	25.58	50.88	0.1	0.68
Select	Part III	27.91	27.73	48.51	0.1	0.68
Our	Part I	16.53	26.62	29.0	0.1	1.05
Superquadric	Part II	11.11	26.38	29.41	0.1	1.2
Representation	Part III	12.17	27.23	29.22	0.1	1.1
Algorithm	Part IV	13.65	25.42	28.01	0.1	1.2

Fig. 7.35 shows superquadric representation results for scene IV containing various objects using the recover-and-select and our proposed algorithms. This scene includes a variety of mechanical parts simulating bin picking tasks. Fig. 7.35(c) indicates that the recover-and-select algorithm fails to recover superquadrics with global deformations. For instance, the handle of the mug and the mechanical part (a tapered object) fail to be represented. In addition, the recover-and-select algorithm cannot recover superquadrics for the objects falling outside the superquadric family. However, for these objects such as the small mechanical parts in the scene, an approximate superquadric representation is more desired than the original mesh representation. The recover-and-select algorithm only recovers 11 superquadrics while the original scene consists of 18 single-part objects. On the other hand, totally 18 superquadrics are obtained since each single-part object is successfully decomposed from the scene, and represented by a superquadric. Superquadrics with global deformations are also successfully recovered from our algorithm. In addition, objects falling outside the superquadric family are represented by superquadrics which approximately represent the size and shape information of the objects. For the computer mouse in the scene, our algorithm recovers a more accurate superquadric than the recover-and-select algorithm since we are able to handle global deformations. Moreover, it takes approximately 25 minutes for the recover-and-select algorithm to recover superquadrics from the scene while our algorithm only needs approximately 5 minutes for part decomposition and 1 minute for superquadric fitting.

Fig. 7.36 shows superquadric representation results for a scene (V). Again, we observe that the recover-and-select algorithm recovers a smaller number of and less accurate superquadrics than our algorithm. In particular, the recover-and-select algorithm fails to recover any superquadric for the crossing-shaped pipe fitting while our algorithm recovers four superquadrics for it. Moreover, it takes approximately 35 minutes for the recover-and-select algorithm to recover superquadrics from the scene while our algorithm only needs approximately 4.5 minutes for part decomposition and 2 minutes for superquadric fitting.

7.1.4 Computational Time for Part Decomposition

For the part decomposition algorithm, Table 7.1 provides a summary of computational time needed for a variety of objects and scenes used in the experiments. The part decomposition time is subject to several factors such as the number of vertices, the number of triangles, and the number of single parts of the object. We use a mug as an example to further investigate the effect of the mesh resolution on the running time of the part decomposition algorithm. The number of triangles for the mug is reduced from 35,000 to 30,000, 25,000, 20,000, 15,000, 10,000, and 5,000 using the mesh reduction function in the RapidForm software (INUS Technology, Inc, 2001). The part decomposition algorithm is next applied to the mug in these various mesh resolutions. The computation platform is an SGI Octane workstation with a single 195 MHz R10000 processor and 128M memory. The running time results are illustrated in Fig. 7.37. An approximately linear relationship exists between the number of triangles and the running time of the part decomposition algorithm.

One thing that needs to be mentioned is that the parameter used in the part decomposition algorithm as the threshold to identify boundaries varies with mesh resolutions. This is due to the fact that curvature estimation is subject to mesh resolutions. As to the computational time

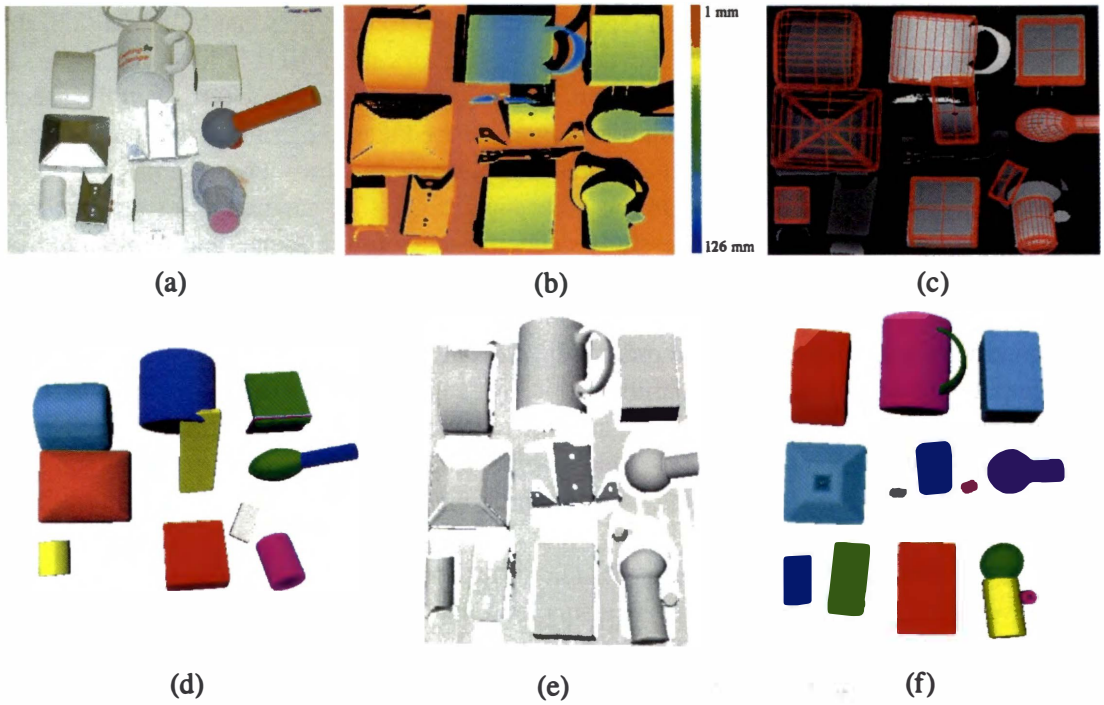


Figure 7.35: Comparison of superquadric representation results for a scene (IV). The scene contains miscellaneous objects simulating a bin picking application. Superquadrics are recovered for the scene using two algorithms including our superquadric representation strategy utilizing part decomposition and the recover-and-select algorithm. The range data is a single-view scan from the IVP Ranger System. The original range image is pseudo-colored to illustrate the range depth variations on the surfaces. Red corresponds to larger distances between the sensor and the object surface while blue corresponds to smaller distances. For the recover-and-select algorithm, a single-view range image is used, and no pre-segmentation is involved. For our method, the mesh consists of 90,237 vertices and 175,298 triangles while the decomposition consists of 18 parts. Since this scan is from a single view, both the range image and the mesh leave many objects as occluded and/or isolated and thus unconnected to other objects. Each of the decomposed parts is fitted to a superquadric model. (a) Photograph of original scene, (b) pseudo-colored range image of original scene, (c) recovered superquadrics using the recover-and-select algorithm and plotted by red wireframes, (d) rendering of superquadrics recovered from the recover-and-select algorithm, (e) rendering of original mesh, and (f) rendering of recovered superquadrics using our superquadric representation algorithm.

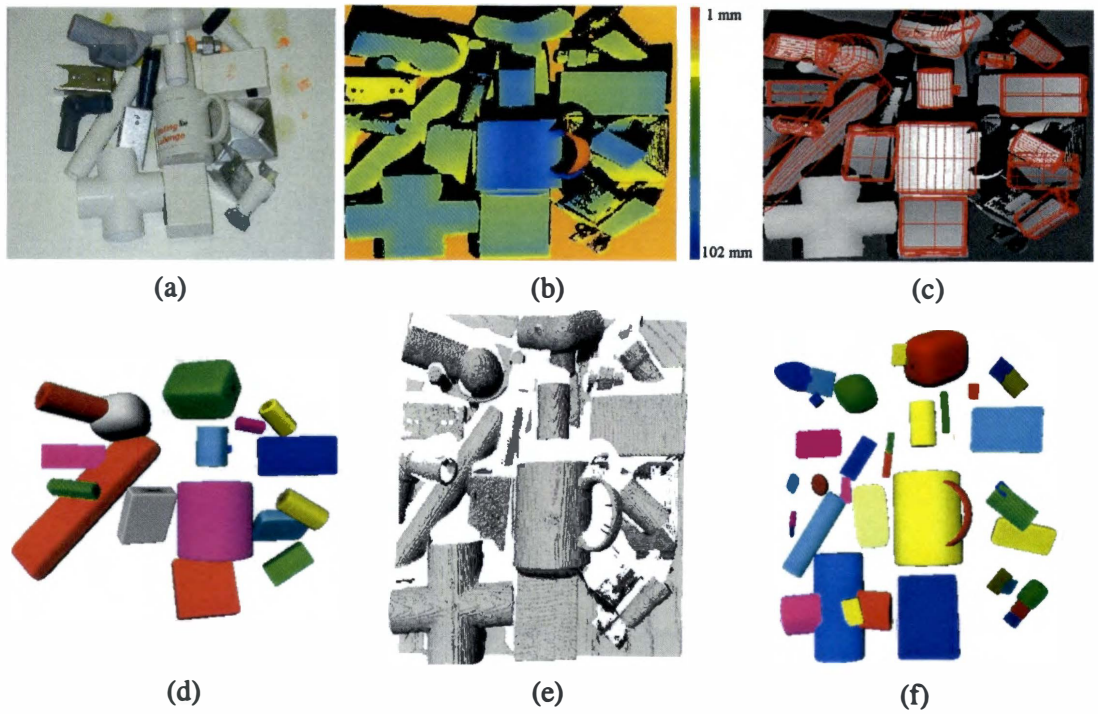


Figure 7.36: Comparison of superquadric representation results for a scene (V). The scene contains miscellaneous objects simulating a bin picking application. Superquadrics are recovered for the scene using two algorithms including our superquadric representation strategy utilizing part decomposition and the recover-and-select algorithm. The range data is a single-view scan from the IVP Ranger System. The original range image is pseudo-colored to illustrate the range depth variations on the surfaces. Red corresponds to larger distances between the sensor and the object surface while blue corresponds to smaller distances. For the recover-and-select algorithm, a single-view range image is used, and no pre-segmentation is involved. For our method, the mesh consists of 89,561 vertices and 169,479 triangles while the decomposition consists of 49 parts. Since this scan is from a single view, both the range image and the mesh leave many objects as occluded and/or isolated and thus unconnected to other objects. Each of the decomposed parts is fitted to a superquadric model. (a) Photograph of original scene, (b) pseudo-colored range image of original scene, (c) recovered superquadrics using the recover-and-select algorithm and plotted by red wireframes, (d) rendering of superquadrics recovered from the recover-and-select algorithm, (e) rendering of original mesh, and (f) rendering of recovered superquadrics using our superquadric representation algorithm.

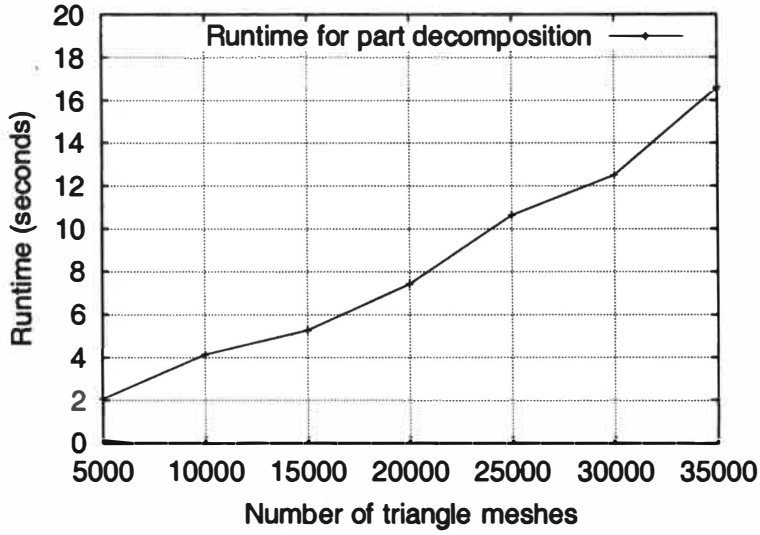


Figure 7.37: Running time analysis for part decomposition of a mug. A variety of mesh resolutions are used.

for superquadric fitting, it takes approximately ten seconds to recover a superquadric from an object consisting of 10, 000 vertices. Therefore, the computational time for superquadric fitting is generally negligible compared to the part decomposition time. Consequently, the overall computational time for our superquadric representation algorithm heavily depends on the part decomposition time.

7.1.5 Remarks

We have tested our part decomposition and superquadric representation algorithms on a large variety of objects and scenes in the experiments. Based on the experimental results we have shown, the following conclusions can be derived for the part decomposition algorithm.

- The proposed part decomposition algorithm demonstrates stable performance on a large variety of objects and scenes with low computational costs and easy implementation.
- The algorithm uses triangle meshes as input data. This is because curvature estimation involved in this algorithm needs neighborhood information contained in the meshes. An extra triangulation or surface reconstruction step is needed when only point clouds are provided.
- The performance of the algorithm is subject to mesh resolution. The resolution of a polygonal mesh determines the amount of surface detail the mesh contains, and is closely related to the number of vertices, edges and faces in the mesh. A coarse resolution mesh contains a small number of vertices while a fine resolution mesh contains a large number

of vertices. The input mesh with a higher resolution enables more accurate boundaries to be identified, and better decomposition results to be obtained with higher computation costs.

- The performance of the algorithm is subject to the smoothness of the input mesh. To obtain better part decomposition results, the meshes of most objects in the experiments are smoothed before they are decomposed. However, although a mesh needs to be overall smooth, the boundaries between linked parts need to be kept instead of being smoothed out so that they can be correctly identified. Therefore, a curvature-based mesh smoothing or a Laplacian smoothing with a bigger weight value is recommended.

- The proposed part decomposition algorithm is able to decompose correct parts in the presence of inaccurate boundaries. This is because the part-growing and the post-processing steps involved in the algorithm can handle inaccurate boundaries.

For the proposed superquadric representation algorithm utilizing part decomposition, the following conclusions can be derived.

- The representation ability of the proposed superquadric representation algorithm is significantly improved by utilizing part decomposition. With our superquadric representation algorithm, complicated, multi-part objects and multi-object scenes can be successfully represented by superquadrics in a two-step manner. The original difficult superquadric representation task is significantly simplified into a data fitting problem.

- Since superquadric fitting is a step subsequent to part decomposition, the superquadric representation results are inevitably affected by the performance of part decomposition. A multi-part object or a multi-object scene needs to be first decomposed into single parts. Each single part can be next fitted to a superquadric model.

- Our algorithm demonstrates overall better performances than the recover-and-select algorithm (Leonardis et al., 1997). First, our algorithm can handle global deformations while the recover-and-select algorithm can only handle regular superquadrics. Second, our algorithm can provide coarse superquadrics for objects outside the superquadric family while the recover-and-select algorithm fails to recover superquadrics for such objects. Finally, our algorithm is approximately five times faster than the recover-and-select algorithm.

One of the most attractive advantages of superquadric representation over triangle meshes is its compactness, i.e., 11 or 15 parameters for a superquadric can represent thousands of triangle meshes. We have demonstrated this characteristic through the experiments with the definition of the data compression ratio DCR. Fig. 7.38 shows a comparison of mesh reduction and superquadric representation for a distributor cap. The original mesh shown in Fig. 7.38(a) is reconstructed from multi-view range scans from the the IVP Ranger System (Integrated Vision Products, 2000), and consists of 58,975 vertices and 117,036 triangles. We iteratively reduce the number of triangles using the mesh decimation function (high quality) in the RapidForm

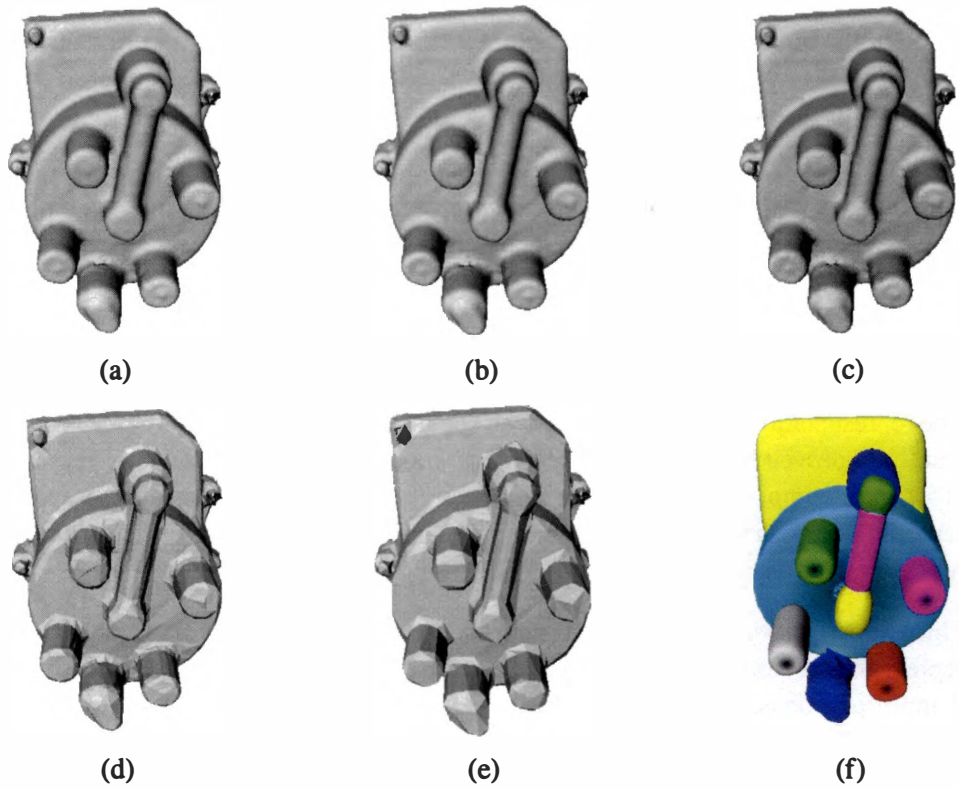


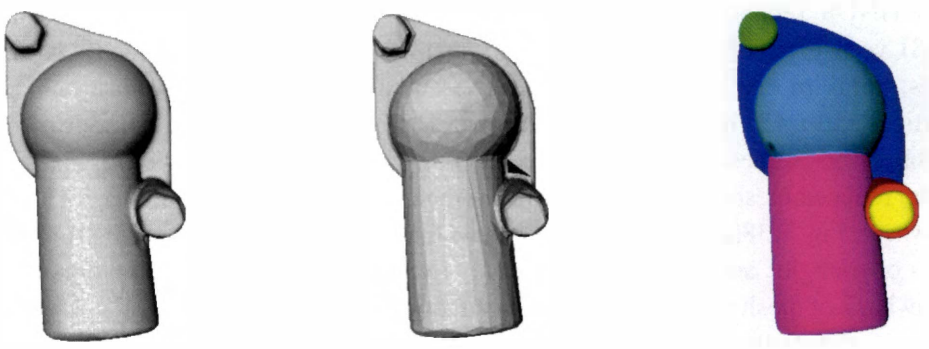
Figure 7.38: Mesh reduction and superquadric representation for a distributor cap. This mesh is a reconstruction from multiple range images using the IVP Ranger System. The original mesh consists of 58,975 vertices and 117,036 triangles. The original mesh is reduced iteratively to four levels by the mesh decimation function (high quality) in the RapidForm software. The original mesh is also decomposed and represented by 13 superquadrics from our algorithm. (a) Rendering of original mesh, (b) rendering of reduced mesh consisting of 29,681 vertices and 58,512 triangles, (c) rendering of reduced mesh consisting of 15,021 vertices and 29,254 triangles, (d) rendering of reduced mesh consisting of 1,767 vertices and 2,925 triangles, (e) rendering of reduced mesh consisting of 971 vertices and 1,455 triangles, and (f) rendering of recovered superquadrics.

software (INUS Technology, Inc, 2001) to four levels as the first one reduced mesh consists of 29,681 vertices and 58,512 triangles; the second one consists of 15,021 vertices and 29,254 triangles; the third one consists of 1,767 vertices and 2,925 triangles; the last one consists of 971 vertices and 1,455 triangles. Fig. 7.38(b) and Fig. 7.38(c) show that little visual difference can be observed between the first and the second reduced meshes and the original mesh since these two reduced meshes still contain huge numbers of vertices and triangles.

However, Fig. 7.38(d) and Fig. 7.38(e) show that rendered surfaces become rough when the numbers of triangles are reduced to 2,925 and 1,455 respectively. This phenomena is a common drawback of mesh reduction, i.e., mesh reduction is always associated by the loss of surface smoothness, especially when the reduction ratio is high (the number of triangles is reduced 40 times for the mesh shown in Fig. 7.38(d) and 80 times for the mesh shown in Fig. 7.38(e)). How to represent a 3D surface model with the least number of vertices and triangles while maintaining the acceptable surface smoothness is an ongoing research area. Superquadric representation addresses this problem in a way different from common mesh reduction approaches. As a volumetric primitive, superquadrics can represent objects compactly, i.e., 11 parameters for a superquadric can represent thousands of triangle meshes. In addition, the explicit definition of superquadrics enables surfaces to be rendered smoothly in 3D. Therefore, superquadric representation is a natural solution to the difficulties in mesh reduction. Fig. 7.38(f) shows 13 recovered and rendered superquadrics for the distributor cap. Superquadrics represent the distributor cap smoothly with only a total of 143 floating numbers while the reduced mesh shown in Fig. 7.38(e) need 7,278 floating numbers. Superquadric representation achieves a DCR of 50.9 for the reduced mesh shown in Fig. 7.38(e) while rendering significantly smoother surfaces. Such characteristics enable superquadrics to significantly benefit many tasks such as scene visualization, object recognition, data transmission, etc.

Fig. 7.39 shows a comparison of mesh reduction and superquadric representation for a water neck. The original mesh shown in Fig. 7.39(a) is reconstructed from multi-view range scans from the the IVP Ranger System (Integrated Vision Products, 2000), and consists of 58,784 vertices and 117,564 triangles. We reduce the number of triangles using the mesh decimation function (high quality) in the RapidForm software (INUS Technology, Inc, 2001) to contain 755 vertices and 1,522 triangles as shown in Fig. 7.39(b). We observe that the rendered surfaces for the reduced mesh are rough compared to the smooth surfaces rendered for the original mesh shown in Fig. 7.39(a).

On the other hand, we have recovered and rendered nine superquadrics for the water neck using our part decomposition and superquadric representation algorithms. The rendered superquadric surfaces shown in Fig. 7.39(c) are as smooth as the original mesh shown in Fig. 7.39(a). However, the superquadric representation for the water neck contains only nine superquadrics, i.e., a total of 99 floating numbers while the original mesh contains 58,784 vertices and 117,564 triangles, i.e, 529,044 floating numbers. Superquadric representation achieves a DCR of 5343.88 over the original triangle mesh representation while still maintaining the same surface smoothness. For the reduced mesh shown in Fig. 7.39(b), it contains 755 vertices and 1,522 triangles, i.e., a total of 6,861 floating numbers. Superquadric representation achieves a DCR of 69.3 over this reduced mesh while rendering significantly smoother surfaces. In summary, superquadric representation demonstrates advantages over common mesh reduction approaches in terms of



(a) (b) (c)

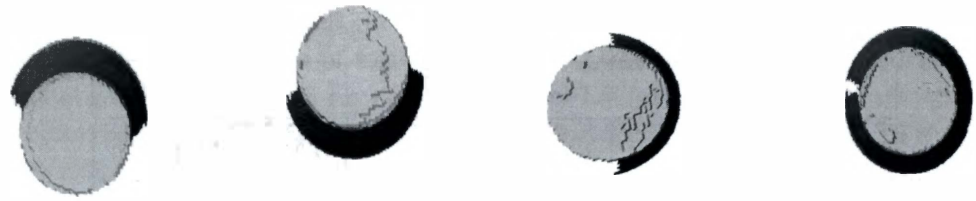
Figure 7.39: Mesh reduction and superquadric representation for a water neck. This mesh is a reconstruction from multiple range images using the IVP Ranger System. The original mesh consists of 58,784 vertices and 117,564 triangles. The original mesh is reduced by the mesh decimation function (high quality) in the RapidForm software to contain 755 vertices and 1,522 triangles. The original mesh is also decomposed and represented by nine superquadrics from our algorithm. (a) Rendering of original mesh, (b) rendering of reduced mesh consisting of 755 vertices and 1,522 triangles, and (c) rendering of recovered superquadrics.

surface smoothness and storage costs. Therefore, our superquadric representation approach can be used as an effective mesh reduction method in applications such as data visualization and transmission.

7.2 View Registration of 3D Surfaces Based on Superquadrics

In a computer vision task involving multi-view scans, the multi-view data are usually acquired and calibrated from independent coordinate systems. View registration aims to align multi-view scans acquired from different coordinate systems into a common coordinate system. This step is indispensable before the multi-view data sets can be further used. Therefore, view registration of 3D surfaces has been a crucial step for computer vision tasks involving multiple views such as view integration, surface reconstruction, etc. Most existing view registration methods need much overlapping to first establish feature correspondences between different views. The rigid transformation between the two views can next be calculated from the correspondences. View registration has been a time-consuming task due to the extensive correspondence search involved.

Our proposed view registration algorithm, however, does not need correspondences between two views, and is therefore, less time-consuming. Therefore, our algorithm does not need much overlapping between the two views. Our view registration algorithm registers two views based on recovered superquadrics from each view. The two views can be correctly registered as long as a superquadric can be recovered for an object which is visible from both views. The proposed view registration algorithm is presented in Chapter 5.



(a) (b) (c) (d)

Figure 7.40: View registration results for a cone. Range images are acquired from three different views using the IVP Ranger System. Superquadrics are first recovered for each view. The three views are then registered in a pairwise manner based on recovered superquadric parameters. (a) Rendering of original mesh from view 1, (b) rendering of original mesh from view 2, (c) rendering of original mesh from view 3, and (d) rendering of registered mesh.

This section provides experimental results for view registration results for 3D surfaces using our view registration algorithm. Superquadrics are first recovered for each view. The rigid transformation between the two views can next be evaluated based on the superquadric parameters. A variety of objects and scenes are used in the experiments. We begin with a single-part object of a cone. Next, an office scene containing three objects including a barrel and two boxes which can be represented by superquadrics is explored. Finally, a scene containing miscellaneous mechanical parts with occlusions is investigated simulating a bin picking application. To demonstrate that our view registration algorithm is independent to the superquadric representation methods, we have used the recover-and-select algorithm to recover superquadrics from the scene.

7.2.1 Single-Part Objects

We first demonstrate our view registration algorithm on a single-part object such as a cone (tapered cylinder). Fig. 7.40 shows view registration results for a cone from three different views. The single-view range images are acquired from the IVP Ranger System. Recovered superquadric parameters from the three views are tabulated in Table 7.6. We observe that accurate superquadrics have been recovered for the object except for a_3 due to single-view scans (partial data) utilized. Based on these superquadric parameters, the three views are registered in a pairwise manner using our algorithm. The registered and merged data shown in Fig. 7.40(d) indicates that the three views are registered correctly despite inaccurate superquadrics recovered. The view registration results demonstrate that our algorithm can register tapered superquadrics, which is a crucial advantage over the original algorithm presented in (Jaklič et al., 2000).

Table 7.6: Recovered superquadric parameters for a cone from three views.

Viewpoint	a_1	a_2	a_3	ε_1	ε_2	k_x	k_y
View 1	19.23	17.42	29.50	0.1	0.97	0.39	0.39
View 2	17.64	19.85	23.20	0.1	0.99	0.38	0.40
View 3	18.59	19.23	26.7	0.1	0.97	0.38	0.39
Ground truth	20	20	20	0.1	1.0	0.4	0.4

7.2.2 Scenes Containing Multiple Objects

Our view registration algorithm can not only register single-part objects but scenes containing multiple objects. The only assumption is that there is at least one object in the scene that can be represented by a superquadric, and the object is visible from all views to be registered. Fig. 7.41 shows an office scene scanned from two views. The range images are acquired from the RIEGL scanner.

From the photographs of the scene scanned from two views as shown in Figs. 7.41(a) and 7.41(d), it can be easily observed that superquadrics can be recovered for three objects in the scene including two boxes and a barrel on the floor from both views. The original 3D point clouds for the three objects scanned from view 1, and corresponding superquadrics are shown in Fig. 7.42. Table 7.7 shows ground truths and recovered superquadric parameters from both views for the three objects in the scene. For the tall box, the fitting error between the recovered superquadric parameters and the original point cloud is 10.85 for view 1 and 16.8 for view 2. Table 7.7 indicates that more accurate superquadrics are recovered for the tall box from view 1. For the barrel, it is observed that accurate superquadric parameters are obtained from both views. For the short box, we observe that the data acquired from view 2 yields a fitting error of 11.32 and 8.64 from view 2. Table 7.7 indicates that more accurate superquadrics are recovered for the short box from view 1. Fig. 7.40 have demonstrated that correct view registration results can be obtained for the cone despite inaccurate superquadrics. For the more complex scene as shown in Fig. 7.41, will the low accuracy of superquadric parameters recovered from view 2 corrupt the view registration or even make it fail? We will see the answer to this question after the view registration experiment.

The recovered superquadrics for the three objects are rendered and put back into the original scene to demonstrate the fitting between the superquadrics and the scene. Fig. 7.43 shows 3D point clouds and recovered superquadrics for the three objects from view 1. We observe that the three recovered superquadrics fit the original scene correctly.

Superquadrics are next recovered for the scene scanned from view 2. Fig. 7.44 shows 3D point clouds and recovered superquadrics for the three objects from view 2. We observe that the three recovered superquadrics fit the original scene well.

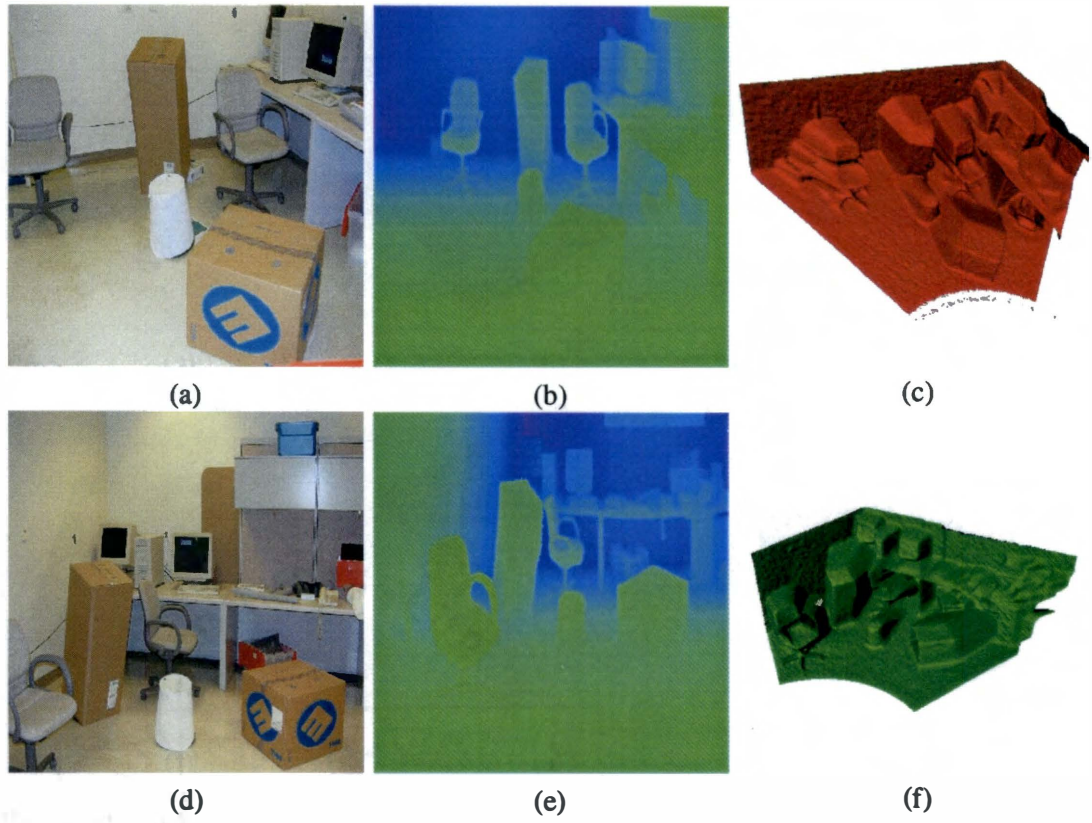


Figure 7.41: An office scene scanned from two views. The range images are acquired by the RIEGL scanner. Superquadrics are recovered for the two boxes and the barrel on the floor, and used in view registration. (a) Photograph of original scene, (b) pseudo-colored range image of the scene, (c) rendering of original mesh from view 1, (d) photograph of original scene, (e) pseudo-colored range image of the scene, and (f) rendering of original mesh from view 2.

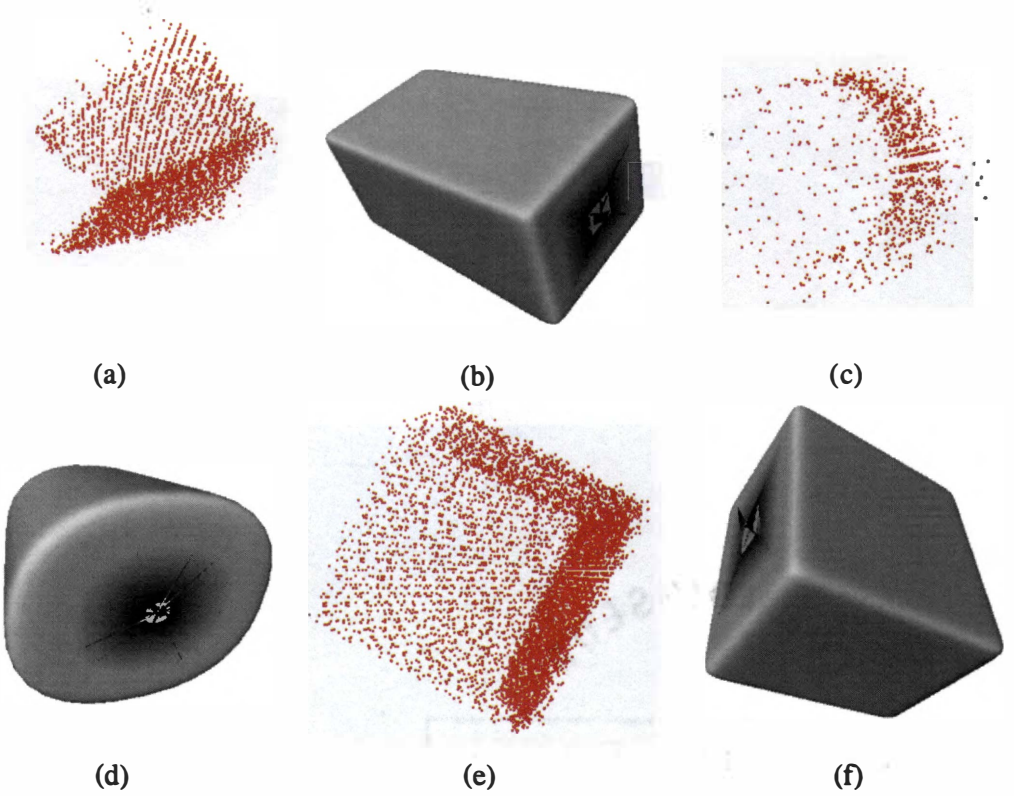
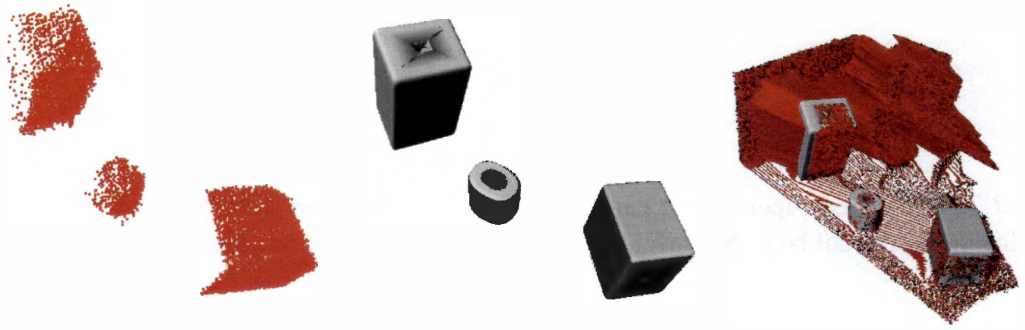


Figure 7.42: Superquadric representation results for three objects in an office scene. The objects including two boxes and a barrel on the floor of the scene are represented by superquadrics. The point cloud for the tall box consists of 3,410 vertices. The barrel consists of 988 vertices. The short box consists of 7,165 vertices. (a) Rendering of 3D point cloud for the tall box, (b) rendering of recovered superquadric for the tall box, (c) rendering of 3D point cloud for the barrel, (d) rendering of recovered superquadric for the barrel, (e) rendering of 3D point cloud for the short box, and (f) rendering of recovered superquadric for the short box.

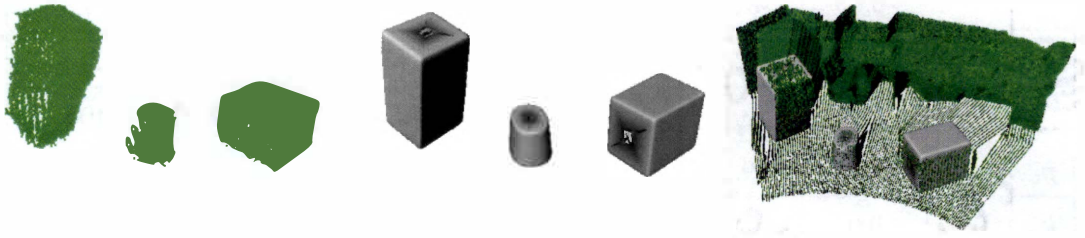
Table 7.7: Recovered superquadric parameters for an office scene from two views. The objects including the tall box, the barrel, and the short box in an office scene are represented by superquadrics.

	Tall box			Barrel			Short box		
	GT	View 1	View 2	GT	View 1	View 2	GT	View 1	View 2
a_1	42.6	42.86	47.22	35.5	37.46	37.86	42.5	44.03	42.8
a_2	52.0	52.89	42.50	30.0	30.16	30.48	61.4	59.67	54.74
a_3	96.5	94.05	92.76	35.5	34.54	34.82	59.5	57.82	60.80
ε_1	0.1	0.1	0.1	0.1	0.14	0.1	0.1	0.1	0.1
ε_2	0.1	0.1	0.1	1.0	0.1	0.87	0.1	0.1	0.1
ϕ		2.99	3.87		1.97	6.50		5.81	5.19
θ		-0.29	-0.28		-0.13	0.08		-1.56	-1.57
ψ		-0.03	4.72		-1.63	-0.12		1.58	1.58
p_x		-361.79	-302.62		-278.21	-265.48		-192.49	-256.84
p_y		-58.39	-73.18		-47.81	1.48		9.40	100.24
p_z		-94.37	-99.94		-132.49	-136.92		-133.141	-135.54
k_x	0.0	0.0	0.0	0.75	0.65	0.74	0.0	0.0	0.0
k_y	0.0	0.0	0.0	0.87	0.85	0.76	0.0	0.0	0.0



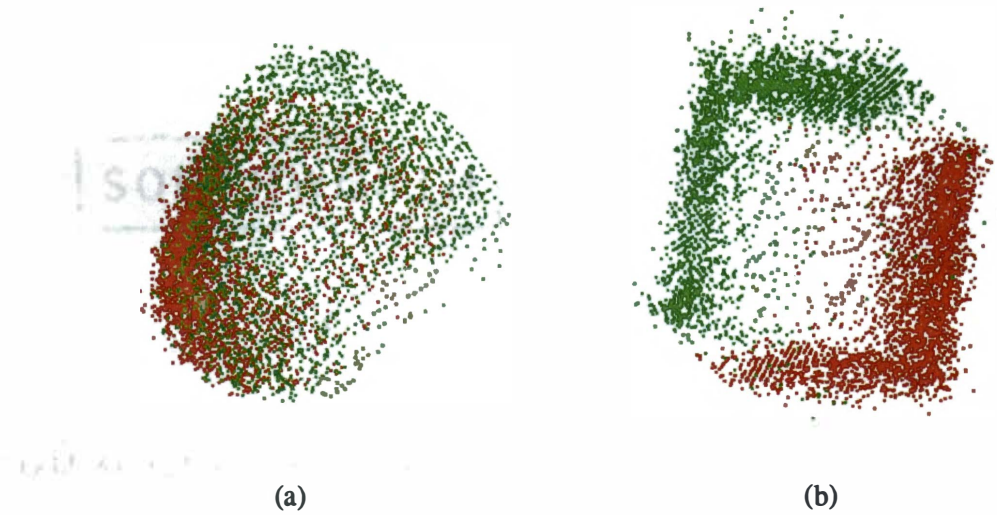
(a) (b) (c)

Figure 7.43: Superquadric representation results for an office scene from view 1. The recovered superquadrics for the two boxes and the barrel on the floor are rendered and plugged back into the original scene to demonstrate the fitting between the superquadrics and the original point clouds. (a) 3D point clouds of the three objects, (b) rendering of recovered superquadrics, and (c) recovered superquadrics plugged back into the original scene.



(a) (b) (c)

Figure 7.44: Superquadric representation results for an office scene from view 2. The recovered superquadrics for the two boxes and the barrel on the floor are rendered and plugged back into the original scene to demonstrate the fitting between the superquadrics and the original point clouds. (a) 3D point clouds of the three objects, (b) rendering of recovered superquadrics, and (c) recovered superquadrics plugged back into the original scene.



(a) (b)

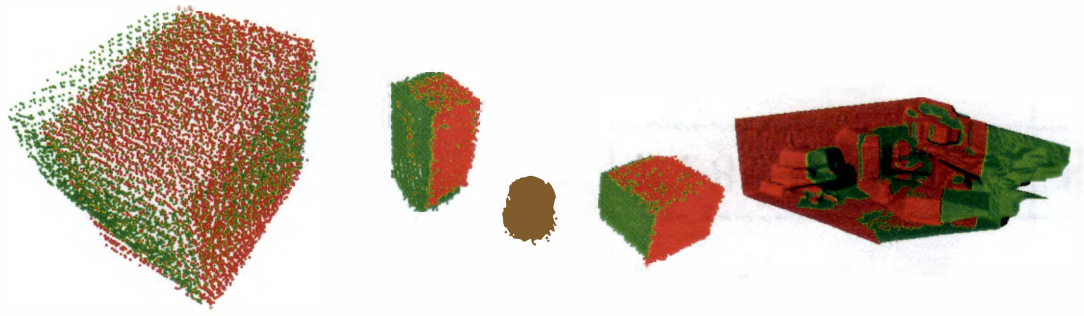
Figure 7.45: View registration results for a tapered barrel. Two view registration algorithms are used. The 3D point cloud in red is obtained from view 1 and the points in green are from view 2. (a) Alignment of the two views using our view registration algorithm and (b) the existing algorithm which cannot handle globally deformed superquadrics.

So far, we have recovered superquadrics for the three objects from both views. To demonstrate the necessity of the extension of our view registration algorithm in terms of handling tapered superquadrics, the two-view data of the barrel are registered using both our and the original algorithms. Fig. 7.45 shows view registration results using the two algorithms. This figure illustrates that our algorithm can register tapered superquadrics while the original registration algorithm fails. Using the three recovered superquadrics, the rigid transformation between the two views is evaluated as

$$T = \begin{bmatrix} 0.609159 & -0.793045 & -0.00232135 & -128.758 \\ 0.792905 & 0.609102 & -0.0172275 & 247.637 \\ 0.0150762 & 0.00865366 & 0.999849 & 0.376007 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

The registration error is evaluated as 1.913.

Based on the view registration result, Fig. 7.46 shows alignment of the short box, the three objects, and the whole scene scanned from the two views. This figure indicates that by registering objects contained in the scene, the whole scene can be correctly registered from our proposed view registration method. In addition, Fig. 7.46(c) demonstrates that our view registration algorithm yields correct registration results despite inaccurate superquadrics. The accuracy of the view registration results can be further refined by the Iterative Closest Point (ICP) algorithm. In summary, we conclude that our proposed view registration algorithm is able to provide correct registration results in the presence of inaccurate superquadric parameters. In



(a) (b) (c)

Figure 7.46: View registration results for an office scene. The data are acquired from two views. (a) The registration of the short paper box as an example, (b) the registration of the three objects which can be represented by superquadrics contained in the scene including a tall paper box, a short paper box, and a small tapered barrel, and (c) the registration of the scene.

other words, the proposed view registration algorithm is robust to noise which causes inaccurate superquadrics being recovered.

A set of experiments for view registration is next performed on a more complex scene consisting of a large variety of shapes and significant occlusions. The objects include blocks, cylinders, ellipsoids, a crank, a distributor cap, a water neck, etc. The single-view range image is scanned from the IVP Ranger System. Three different views of the scene are scanned. The recover-and-select algorithm (Leonardis et al., 1997) is used to automatically recover superquadrics from each single-view range image. Fig. 7.47 shows superquadric representation results for the scene (VI) scanned from view 1. The 3D calibrated range data shown in Fig. 7.47(b) is rendered in red to distinguish from scans acquired from other different views. Recovered superquadrics are depicted in Fig. 7.47(d) as red wireframes. It can be observed that correct superquadrics have been recovered for the blocks, the ellipsoids, the cone, and the handle of the water neck. The remaining objects in the scene fail to be represented due to either irregular shapes or serious occlusions.

Fig. 7.48 shows superquadric representation results for the scene (VI) scanned from view 2. The 3D calibrated range data shown in Fig. 7.47(b) is rendered in green to distinguish from scans acquired from other different views. Recovered superquadrics are depicted in Fig. 7.48(d) as red wireframes. It can be observed that similar superquadric representation results have been obtained from this view (view 2) as from view 1. In particular, correct superquadrics are recovered for the blocks, the ellipsoids, the cone, and the handle of the water neck. The remaining objects in the scene fail to be represented due to either irregular shapes or serious occlusions.

Fig. 7.49 shows superquadric representation results for the scene (VI) scanned from view 3. The 3D calibrated range data shown in Fig. 7.47(b) is rendered in blue to distinguish from scans acquired from other different views. Recovered superquadrics are depicted in Fig. 7.49(d)

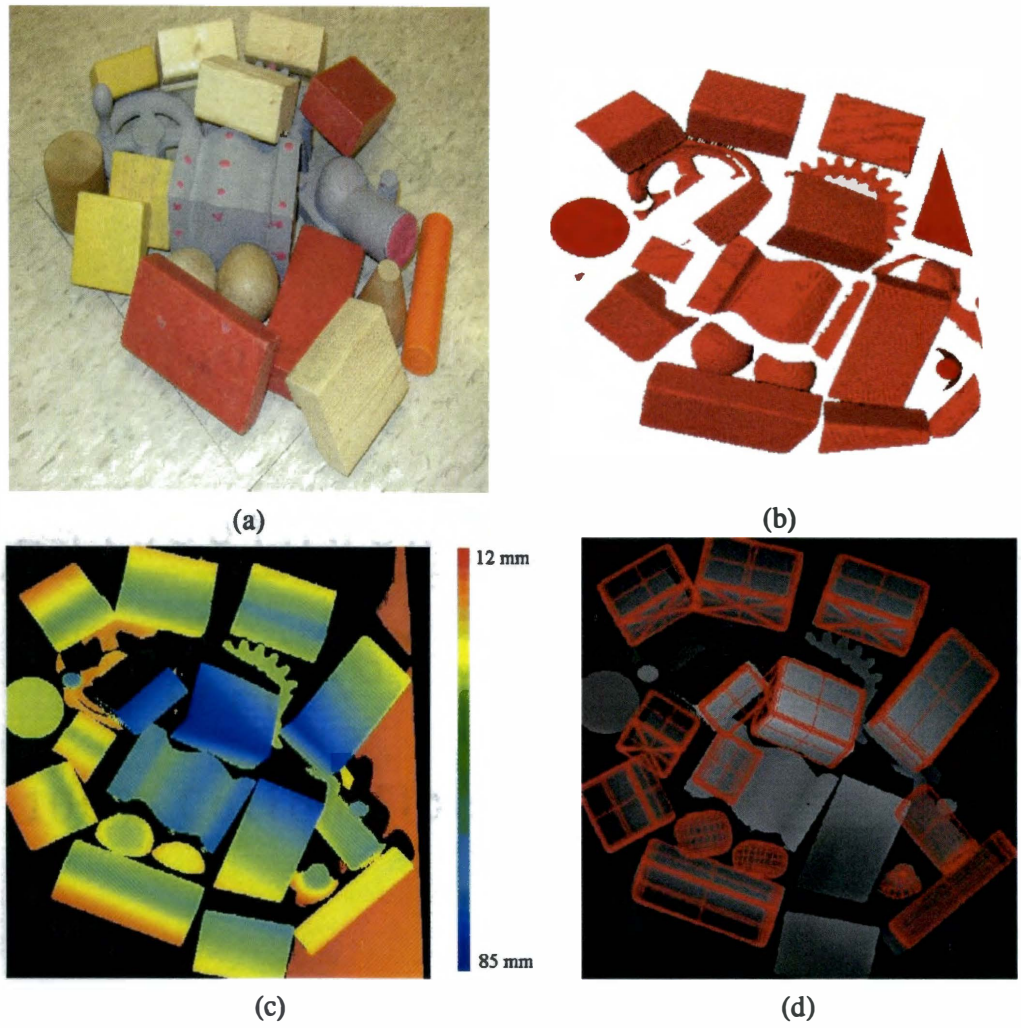


Figure 7.47: Superquadric representation results for a scene (VI) from view 1. The scene consists of miscellaneous objects simulating a bin picking task. The range image is acquired from the IVP Ranger System. The original range image is pseudo-colored to illustrate the range depth variations on the surfaces. Red corresponds to larger distances between the sensor and the object surface while blue corresponds to smaller distances. The recover-and-select algorithm is used to recover superquadrics from the scene. (a) Photograph of original scene, (b) rendering of original mesh, (c) pseudo-colored range image of original scene, and (d) recovered superquadrics depicted as red wireframes.

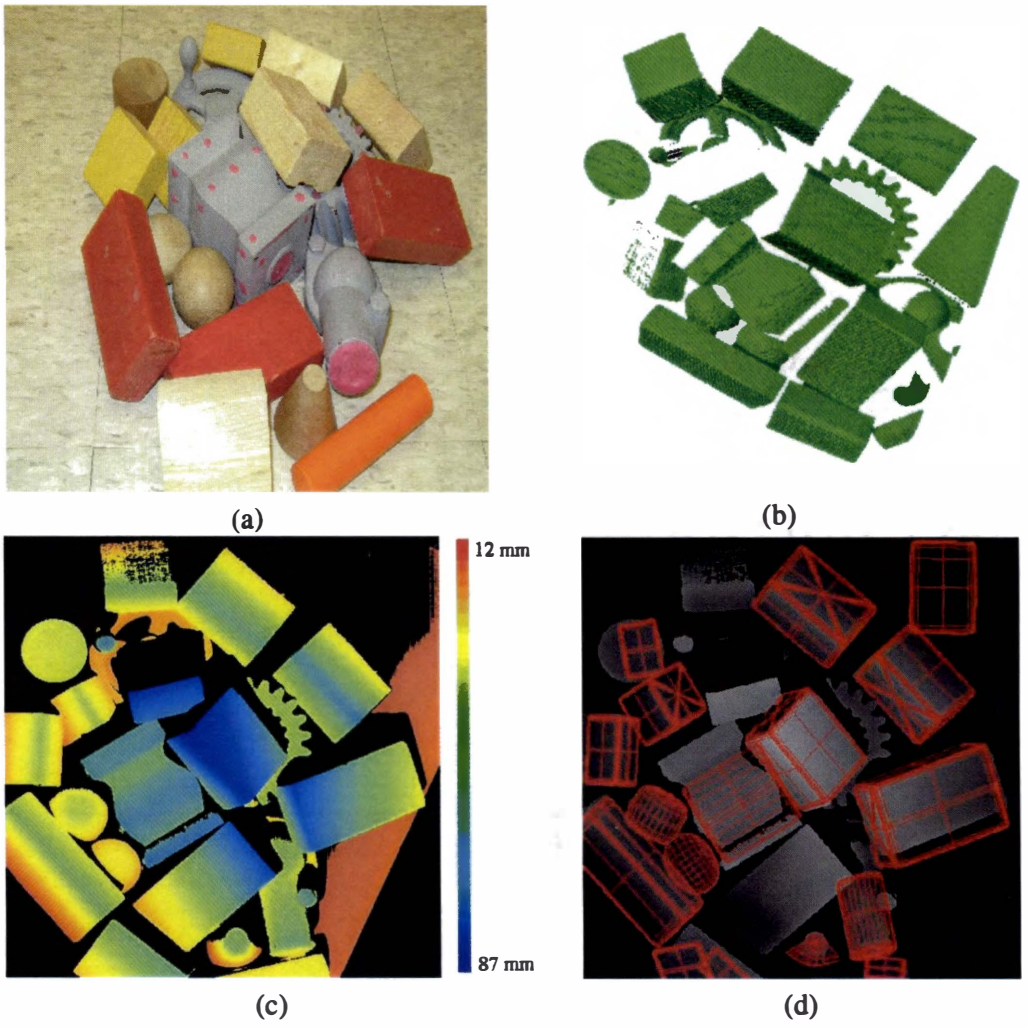


Figure 7.48: Superquadric representation results for a scene (VI) from view 2. The scene consists of miscellaneous objects simulating a bin picking task. The range image is acquired from the IVP Ranger System. The original range image is pseudo-colored to illustrate the range depth variations on the surfaces. Red corresponds to larger distances between the sensor and the object surface while blue corresponds to smaller distances. The recover-and-select algorithm is used to recover superquadrics from the scene. (a) Photograph of original scene, (b) rendering of original mesh, (c) pseudo-colored range image of original scene, and (d) recovered superquadrics depicted as red wireframes.

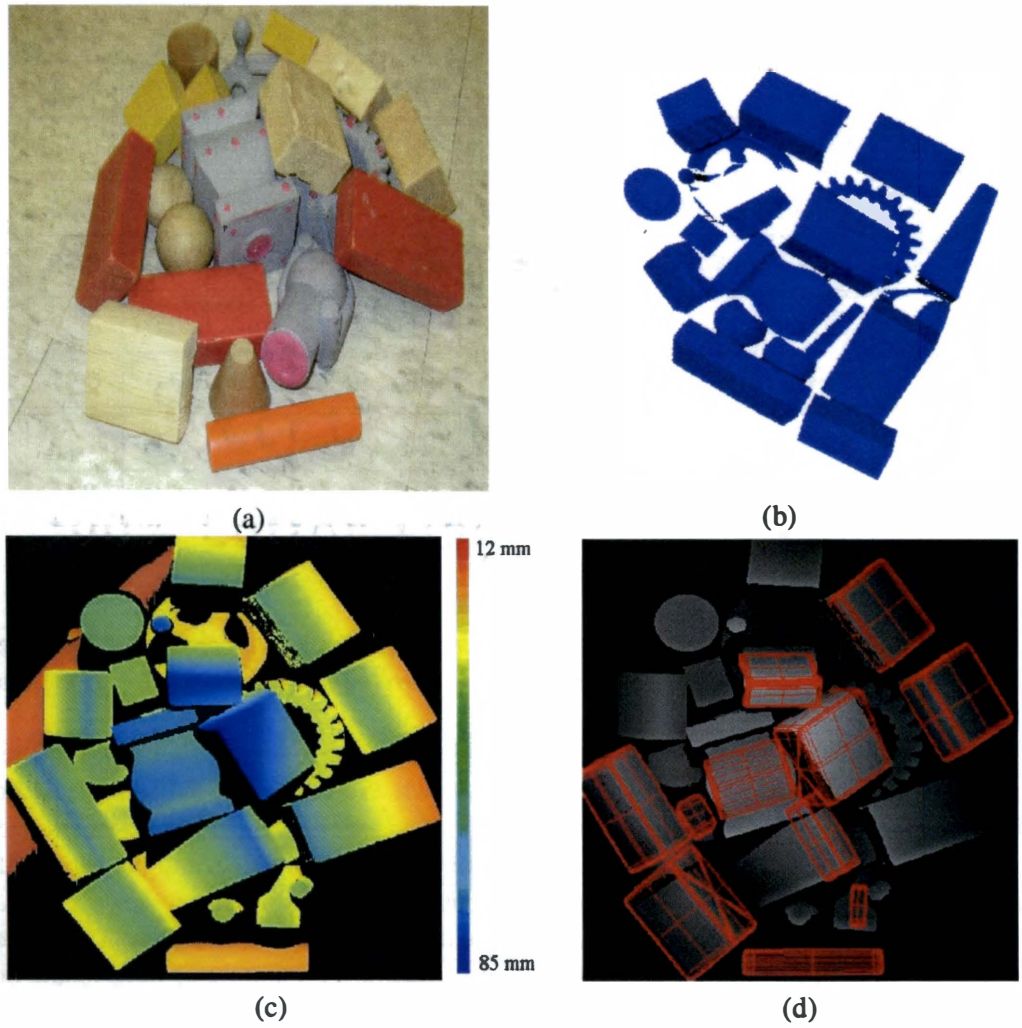


Figure 7.49: Superquadric representation results for a scene (VI) from view 3. The scene consists of miscellaneous objects simulating a bin picking task. The range image is acquired from the IVP Ranger System. The original range image is pseudo-colored to illustrate the range depth variations on the surfaces. Red corresponds to larger distances between the sensor and the object surface while blue corresponds to smaller distances. The recover-and-select algorithm is used to recover superquadrics from the scene. (a) Photograph of original scene, (b) rendering of original mesh, (c) pseudo-colored range image of original scene, and (d) recovered superquadrics depicted as red wireframes.

Table 7.8: Recovered superquadric parameters for a scene (VI) from two views. Both view 1 and view 2 are used.

Object	View	a_1	a_2	a_3	ε_1	ε_2	k_x	k_y	Error
Block 1	View 1	26.18	56.98	14.96	0.1	0.1	0.0	0.0	0.84
	View 2	13.40	29.41	55.39	0.1	0.11	0.0	0.0	0.78
Block 2	View 1	55.90	26.34	16.57	0.1	0.1	0.0	0.0	1.02
	view 2	39.69	57.69	14.15	0.11	0.1	0.0	0.0	0.85
Handle	View 1	13.25	12.09	23.45	0.11	0.95	0.0	0.0	0.89
	View 2	13.45	12.60	22.95	0.10	0.92	0.0	0.0	0.91
Cone	View 1	9.14	25.63	23.46	0.1	1.02	0.38	0.41	1.15
	View 2	22.96	9.58	27.34	0.1	1.0	0.42	0.41	1.09

as red wireframes. We observe that much less superquadrics have been recovered from view 3 than from previous views. Superquadrics are only recovered for the blocks due to significant occlusions.

After superquadrics are recovered for each single-view range image, the view registration algorithm is used to pairwisely register the three views. Superquadrics with the least fitting errors are selected for the view registration. To register view 1 and view 2, superquadrics recovered for the two biggest blocks, the cone at the bottom of the scene, and the handle of the water neck are used to evaluate the rigid transformation between these two views. Table. 7.8 shows recovered parameters for the four superquadrics from view 1 and view 2. Compared with ground truth values, these recovered superquadrics are accurate and have small fitting errors. The pose information about these superquadrics are not listed, but they are used for view registration. The transformation matrix between view 1 and view 2 is calculated as

$$T_{12} = \begin{bmatrix} 0.990767 & -0.132262 & 0.0297938 & 30.4517 \\ 0.126852 & 0.981909 & 0.140583 & 7.40802 \\ -0.04784867 & -0.135505 & 0.98962 & 25.914 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

To register view 1 and view 3, superquadrics recovered for the two blocks to the right of the gear and the block on top of the crank are used. Table. 7.9 shows recovered parameters for the three superquadrics from view 3. Compared with ground truth values, these recovered superquadrics are accurate and have small fitting errors. The transformation matrix between view 1 and view 3 is calculated as

Table 7.9: Recovered superquadric parameters for a scene (VI) from view 3.

Object	a_1	a_2	a_3	ε_1	ε_2	Error
Block 1	45.81	24.20	52.48	0.1	0.1	0.99
Block 2	27.32	51.47	54.50	0.12	0.1	0.87
Block 3	6.54	18.51	44.40	0.1	0.14	1.0

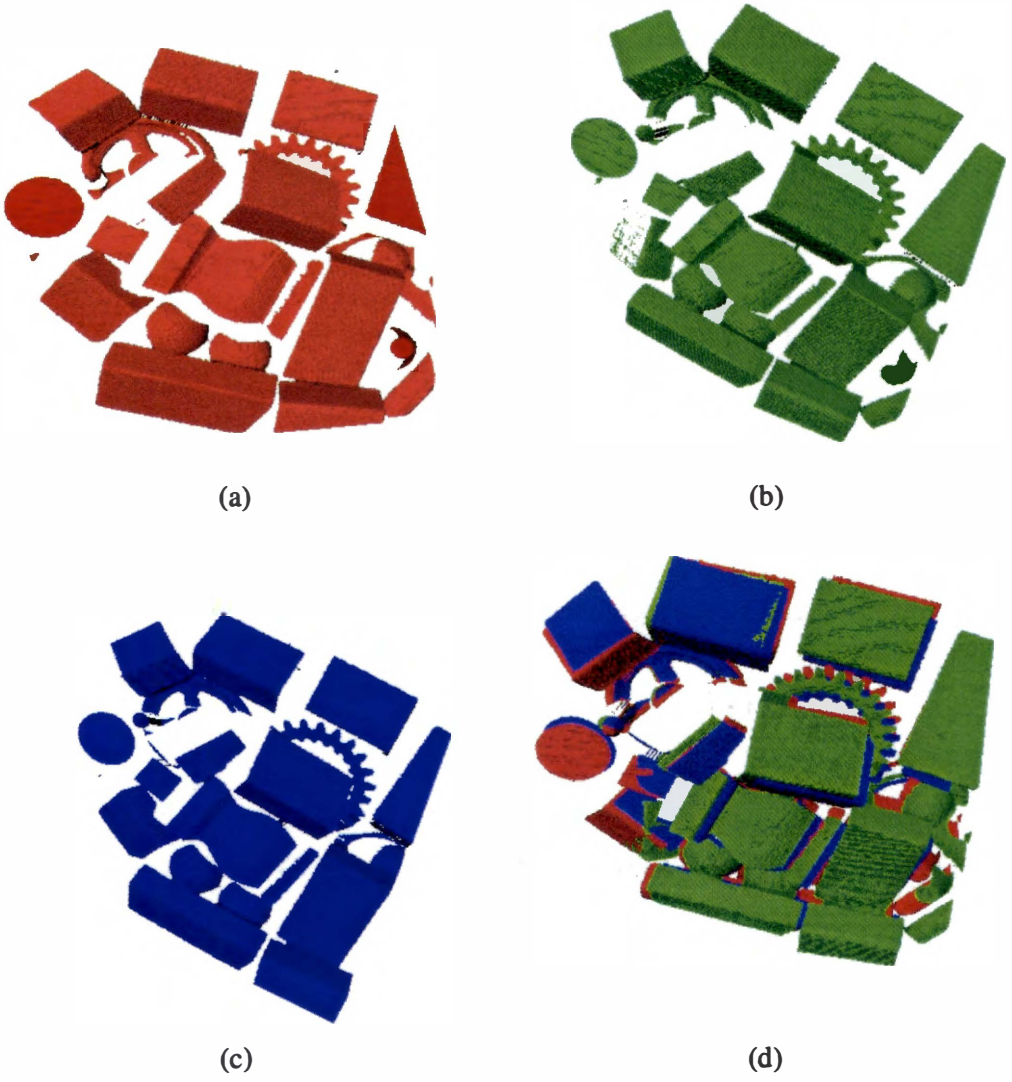
$$T_{13} = \begin{bmatrix} 0.96217 & -0.271681 & -0.0204292 & 78.2011 \\ 0.272353 & 0.961108 & 0.045771 & 54.7197 \\ 0.00719948 & -0.0496035 & 0.998743 & 10.7555 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

The three views are next registered into a common coordinate system based on the obtained pairwise view registration results. Fig. 7.50 shows view registration results for the scene from the three views. The registered and merged 3D mesh for the scene from the three views is shown in Fig. 7.50(d). It can be observed that the three views are correctly registered with small misalignments using our view registration algorithm. It is concluded that our view registration algorithm is able to successfully register complex scenes despite inaccurate superquadrics. The misalignments contained in the view registration results as shown in Fig. 7.50(d) can be reduced, and the view registration results can be further refined by the ICP algorithm (Zhang, 1994).

7.2.3 Remarks

Based on the experimental results demonstrated in this section, the following conclusions on the proposed view registration algorithm based on superquadric representation are summarized.

- The proposed view registration algorithm is an extended version of an existing range image registration algorithm proposed in (Jaklič et al., 2000).
- We have extended the view registration algorithm from 2.5D range image to 3D point clouds. The advantage of this extension is that registered and integrated 3D point clouds can significantly improve superquadric representation in terms of both confidence and accuracy of recovered parameters. In contrast, 2.5D registered and integrated range images have no such benefit.
- We have extended the view registration algorithm to handle tapered superquadrics by deriving a set of new equations to evaluate moments for tapered superquadrics. The original algorithm (Jaklič et al., 2000) can only handle undeformed superquadrics. The motivation of this extension is that tapered objects are common in the real world, and need to be represented by tapered superquadrics. We demonstrate the necessity of this



(a) (b)

(c) (d)

Figure 7.50: View registration results for a scene (VI) from three views. (a) Rendering of original mesh from view 1, (b) rendering of original mesh from view 2, (c) rendering of original mesh from view 3, and (d) view registration results for the scene from three views.

extension through experiments by showing that the existing algorithm fails to register tapered superquadrics but our extended algorithm succeeds.

- We have extended the view registration algorithm to register complex scenes containing multiple objects and backgrounds. The only assumption is that there is at least one object in the scene that can be represented by a superquadric, and the object is visible from all views to be registered.

- The proposed view registration algorithm is insensitive to inaccurate superquadrics recovered from noisy data. The registration result provided by the algorithm is accurate enough to serve as an initial estimate for the Iterative Closest Point (ICP) algorithm to achieve higher registration accuracy. Through the proposed view registration algorithm, superquadrics demonstrate versatilities and wider applications to computer vision tasks.

7.3 Multi-View Superquadric Representation of Scenes

We propose a multi-view superquadric representation algorithm to address two problems in the research literature of superquadric representation. By incorporating both part decomposition and multi-view range data, the proposed algorithm is able to not only represent multi-part objects or multi-object scenes but also achieve high confidence and high accuracy of recovered superquadrics. The abilities of representing complex objects and scenes while achieving high confidence of recovered superquadrics are major advantages of the proposed multi-view superquadric representation algorithm. Recalling the diagram of this algorithm shown in Fig. 3.7, the multi-view superquadric representation algorithm consists of (i) part decomposition and initial superquadric model recovery from single-view range data, (ii) pairwise view registration based on recovered superquadric models, (iii) view integration of multi-view scans, (iv) part decomposition of integrated data, and (v) final superquadric fitting for each decomposed part.

We have shown experimental results for part decomposition and view registration in previous sections. In this section, we complete the multi-view superquadric representation pipeline by presenting experimental results for each step within the pipeline. A wide variety of objects and scenes are used in the experiments. Range images for the objects and scenes are acquired using the IVP Ranger System. Single-part objects and scenes containing multiple objects are explored separately. We begin with a single-part object of a cone scanned from three views. Next, a set of scenes consisting of multiple objects with various geometries and occlusions are investigated to simulate waste area clean-up and bin picking applications. For the scenes, we have used four or five views to demonstrate our multi-view superquadric representation algorithm. The data sets we have explored in the experiments can be classified into the following categories.

- Single-part objects. A cone is used in the experiments as a simple example of a single-part object. In addition, the cone needs to be represented by a tapered superquadric, which shows the contribution of our view registration algorithm on tapered superquadrics.

- Four scenes consisting of objects with various geometries. To start from simple, the first two scenes contain two separate objects. The third scene contains 14 objects including

Table 7.10: Recovered superquadric parameters for a cone. Three single views and integrated data are used.

Data	a_1	a_2	a_3	ε_1	ε_2	k_x	k_y
View 1	19.23	17.42	29.50	0.1	0.97	0.39	0.39
View 2	17.64	19.85	23.20	0.1	0.99	0.38	0.40
View 3	18.59	19.23	26.7	0.1	0.97	0.38	0.39
Integrated data	19.55	18.42	19.85	0.1	0.98	0.39	0.40
Ground truth	20	20	20	0.1	1.0	0.4	0.4

blocks, cones, spheres, cylinders, a water neck, and a distributor cap with occlusions. The fifth scene is the most complicated regarding the occlusions and the variety of the objects it contains. The fifth scene contains more than 30 objects including a crossing-shaped pipe fitting, a water neck, a computer mouse, a cup, a crossing-shaped pipe fitting, small bolts, power adapters, PVC pipes, mechanical parts, etc. with significant occlusions. This scene aims to simulate waste area clean-up and bin picking applications.

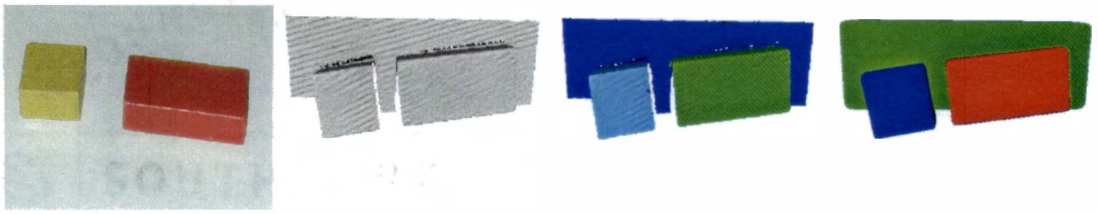
7.3.1 Single-Part Objects

Fig. 7.40 shows view registration and integration results for a cone from three different views. The single-view range images are acquired from the IVP Ranger System. Figs. 7.40(a), 7.40(b) and 7.40(c) show 3D calibrated range data (triangle meshes) of the cone scanned from three views. Using our view registration algorithm, the three views are registered after superquadrics are recovered from each view. Fig. 7.40(d) shows the registered and integrated data of the three views. We demonstrate the correctness of the view registration results. The smaller bottom surfaces scanned from the three views are correctly registered, and the integrated data contains more surfaces of the cone than single view scans. We calculate the fitting error percentage P_{err} for each parameter by

$$P_{err} = \frac{|GT - RV|}{GT} \times 100\%, \tag{7.2}$$

where GT denotes ground truth value of a parameter and RV denotes corresponding recovered value.

To demonstrate the efficiency of the proposed multi-view superquadric representation algorithm quantitatively, we show ground truths for the cone, and recovered superquadric parameters from three single views and the integrated data in Table 7.10. We observe that parameters a_1, a_2 and a_3 recovered from the three views are significantly different from each other. Therefore, it is difficult to determine parameters recovered from which view are more convincing and accurate when ground truths are unavailable. This demonstrates that superquadrics recovered from



(a) (b) (c) (d)

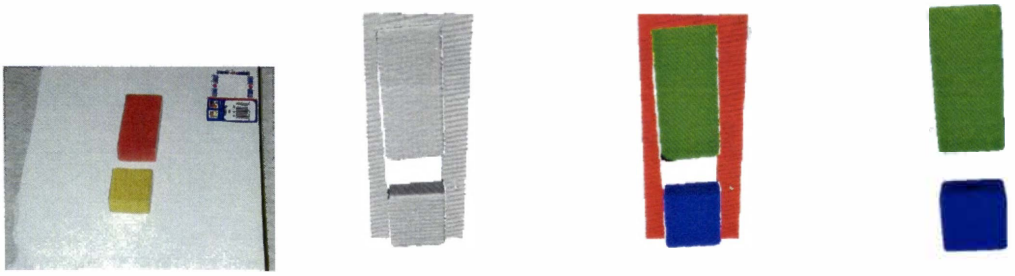
Figure 7.51: Superquadric representation results for a scene (I) from view 1. The scene contains two separate blocks. This mesh is a single-view scan from the IVP Ranger System. The mesh consists of 20,156 vertices and 39,157 triangles while the decomposition consists of three parts including the background. Each decomposed part is fitted to a superquadric model. Superquadrics are rendered in 3D using quad meshes based on recovered superquadric parameters. (a) Photograph of the original scene, (b) rendering of original mesh, (c) decomposed parts labeled in different colors, and (d) rendering of recovered superquadrics.

single-view data have low confidence. In contrast, superquadrics recovered from integrated data have higher confidence and accuracy due to more input data utilized. For instance, compared with the ground truth of parameter a_3, the integrated data recovers a significantly more accurate parameter than any of the three single views. The fitting error percentage P_{err} for a_3 is evaluated as 0.75% from the integrated data, 47.5% from view 1, 16% from view 2, and 33.5% from view 3. There is no significant difference between single-view and multi-view recovery results for the remaining parameters. We conclude that the proposed multi-view superquadric representation algorithm recovers more accurate superquadrics than single-view recovery strategies.

7.3.2 Scenes Containing Multiple Objects

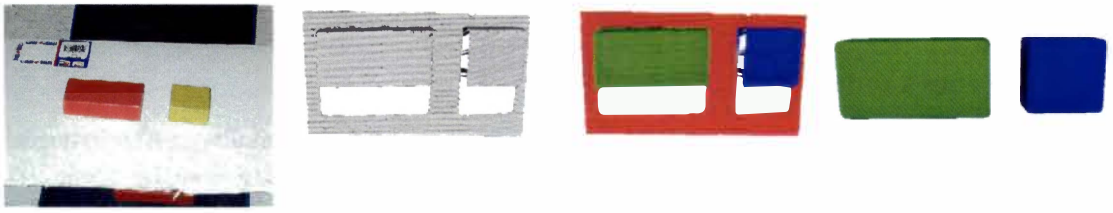
This section presents experimental results on multi-object scenes for our multi-view superquadric representation algorithm. The four scenes used in this section correspond to scene (I), scene (II), scene (III), and scene (V) in the experiments of part decomposition presented in Section 7.1. For scene (III), five views are used for multi-view superquadric representation. Four views are used for the other three scenes. Single-view range images are acquired from the IVP Ranger System. To begin with simpler examples, scene (I) and scene (II) only contain two separate objects. The other scenes contain more objects with more occlusions simulating bin picking applications.

Figs. 7.51 through 7.54 show part decomposition and superquadric representation results for a scene (I) from four views. This scene contains two separate blocks as a simple example. For each of the four views, three single parts including the two blocks and the background are decomposed from the scene as shown in Figs. 7.51(c) through 7.54(c). For the decomposed parts in the scene, we recover superquadrics for the two blocks for each single-view scan.



(a) (b) (c) (d)

Figure 7.52: Superquadric representation results for a scene (I) from view 2. The scene contains two separate blocks. This mesh is a single-view scan from the IVP Ranger System. The mesh consists of 15,551 vertices and 29,375 triangles while the decomposition consists of three parts including the background. Each decomposed part is fitted to a superquadric model. Superquadrics are rendered in 3D using quad meshes based on recovered superquadric parameters. (a) Photograph of the original scene, (b) rendering of original mesh, (c) decomposed parts labeled in different colors, and (d) rendering of recovered superquadrics.



(a) (b) (c) (d)

Figure 7.53: Superquadric representation results for a scene (I) from view 3. The scene contains two separate blocks. This mesh is a single-view scan from the IVP Ranger System. The mesh consists of 19,687 vertices and 37,911 triangles while the decomposition consists of three parts including the background. Each decomposed part is fitted to a superquadric model. Superquadrics are rendered in 3D using quad meshes based on recovered superquadric parameters. (a) Photograph of the original scene, (b) rendering of original mesh, (c) decomposed parts labeled in different colors, and (d) rendering of recovered superquadrics.

(a) (b) (c) (d)

Figure 7.54: Superquadric representation results for a scene (I) from view 4. The scene contains two separate blocks. This mesh is a single-view scan from the IVP Ranger System. The mesh consists of 13,099 vertices and 24,558 triangles while the decomposition consists of three parts including the background. Each decomposed part is fitted to a superquadric model. Superquadrics are rendered in 3D using quad meshes based on recovered superquadric parameters. (a) Photograph of the original scene, (b) rendering of original mesh, (c) decomposed parts labeled in different colors, and (d) rendering of recovered superquadrics.

The recovered superquadric parameters for the two blocks scanned from the four views, and the corresponding ground truths are tabulated in Table 7.11. From the recovered and rendered superquadrics shown in Figs. 7.51(d) through 7.54(d), we observe visual differences of the recovered superquadrics among the four views. Table 7.11 verifies this difference quantitatively. For instance, for the bigger block in the scene, significantly different superquadrics are recovered for the four single views. This demonstrates the weakness caused by single-view superquadric representation. In such cases, superquadrics recovered from single views are subject to viewpoints from which the scan is acquired. Without ground truths as priori knowledge about the scene, the confidence of superquadrics recovered from single views remain unknown. In addition, Table 7.11 shows that none of the four views recovers accurate superquadrics compared with the ground truths. In summary, the experiments demonstrate that superquadrics recovered from single views have low confidence and accuracy. To solve this problem, we apply our multi-view superquadric representation strategy to this scene.

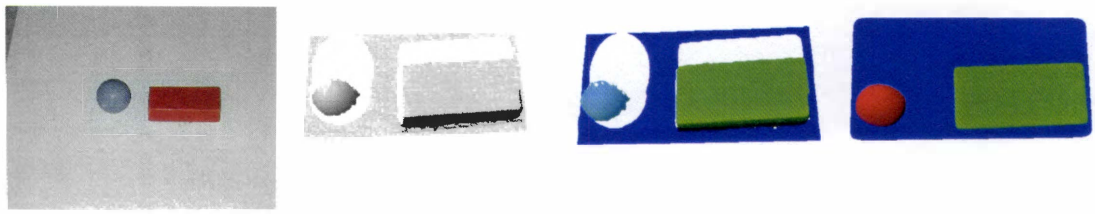
Fig. 7.55 shows registered and integrated data of the scene from the four views, the part decomposition results, and the superquadric representation of the integrated data. This figure demonstrates that the four views are correctly registered using our view registration algorithm. The registered data is integrated using the RapidForm software (INUS Technology, Inc, 2001). The integrated data contains more data and/or surfaces than any single-view scan. As shown in Table 7.11, the superquadric parameters recovered from the integrated data are more accurate than those from any of the four single views when compared with the ground truths. In addition, superquadrics recovered from the integrated data have higher confidence due to more complete input data utilized. Consequently, we conclude that our multi-view superquadric representation algorithm is able to recover superquadrics with higher confidence and accuracy than single-view superquadric representation approaches.

Table 7.11: Recovered superquadric parameters for a scene (I). Two blocks in the scene are represented by superquadrics. Four single views as well as integrated data are used.

Object	Data	a_1	a_2	a_3	ε_1	ε_2
Bigger Block	**Ground truth**	**14.0**	**28.0**	**58.5**	**0.1**	**0.1**
	View 1	14.98	33.48	61.04	0.1	0.17
	View 2	14.99	28.16	59.13	0.1	0.1
	View 3	17.41	31.79	61.14	0.1	0.19
	View 4	10.85	27.71	58.28	0.1	0.1
	Integrated data	**14.85**	**28.86**	**58.79**	**0.1**	**0.1**
Smaller Block	**Ground truth**	**14.0**	**34.5**	**34.5**	**0.1**	**0.1**
	View 1	15.24	31.39	33.62	0.12	0.1
	View 2	15.59	29.09	30.04	0.14	0.1
	View 3	17.26	30.1	30.68	0.1	0.16
	View 4	15.74	30.68	30.9	0.14	0.14
	Integrated data	**14.99**	**33.6**	**34.87**	**0.1**	**0.1**

(a) (b) (c)

Figure 7.55: Multi-view superquadric representation results for a scene (I). The original scene is scanned from four different views. Superquadrics are first recovered from each single view. The four-view data are registered based on the recovered superquadric parameters. The registered multi-view data are integrated using the RapidForm software. The integrated mesh consists of 43,007 vertices and 83,855 triangles while the decomposition consists of three parts including the background. The integrated data are decomposed into single parts. Each decomposed part is fitted to a superquadric model. Superquadrics are rendered in 3D using quad meshes based on recovered superquadric parameters. (a) Rendering of registered and integrated mesh of the original scene scanned from four views, (b) decomposed parts labeled in different colors, and (c) rendering of recovered superquadrics.



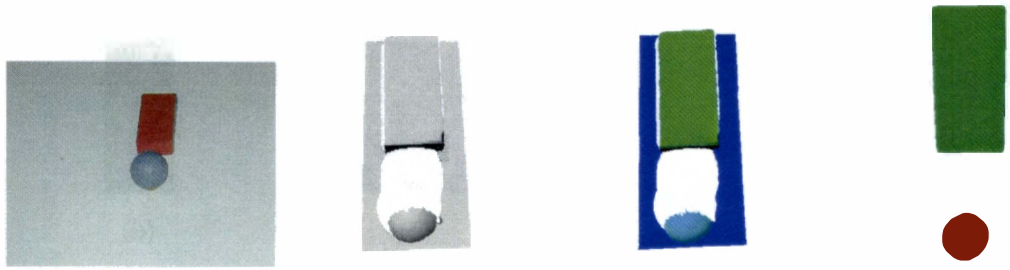
(a) (b) (c) (d)

Figure 7.56: Superquadric representation results for a scene (II) from view 1. The scene contains two separate objects including a block and a sphere. This mesh is a single-view scan from the IVP Ranger System. The mesh consists of 27,615 vertices and 53,266 triangles while the decomposition consists of three parts including the background. Each decomposed part is fitted to a superquadric model. Superquadrics are rendered in 3D using quad meshes based on recovered superquadric parameters. (a) Photograph of the original scene, (b) rendering of original mesh, (c) decomposed parts labeled in different colors, and (d) rendering of recovered superquadrics.

Figs. 7.56 through 7.59 show part decomposition and superquadric representation results for a scene (II) from four views. This scene contains two separate objects including a block and a sphere. For each of the four views, three single parts including the block, the sphere, and the background are decomposed from the scene as shown in Figs. 7.56(c) through 7.59(c). For the decomposed parts in the scene, we recover superquadrics for the block and the sphere for each single-view scan.

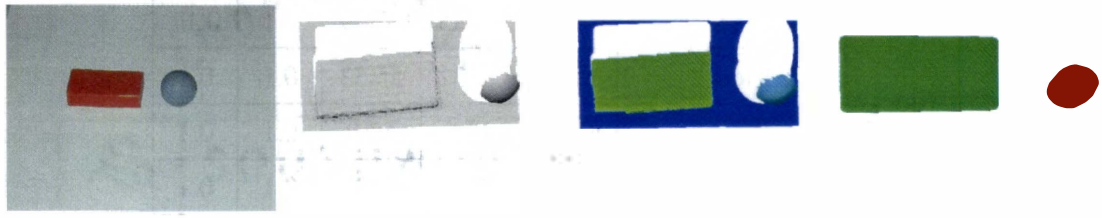
The recovered superquadric parameters for the two objects from the four views and the corresponding ground truths are tabulated in Table 7.12. From the recovered superquadrics shown in Figs. 7.56(d) through 7.59(d), we observe visual differences of the recovered superquadrics among the four views. Table 7.12 verifies this difference quantitatively. For instance, for the sphere in the scene, significantly different superquadrics are recovered for the four single views. This demonstrates the weakness caused by single-view superquadric representation. In such cases, superquadrics recovered from single views are subject to viewpoints from which the scan is acquired. Superquadrics recovered from single views tend to be inaccurate due to partial data used. Without ground truths as priori knowledge about the scene, the confidence of superquadrics recovered from single views remain unknown. In addition, Table 7.12 shows that none of the four views recovers accurate superquadrics compared with the ground truths. In summary, the experiments demonstrate that superquadrics recovered from single-view superquadric representation strategies have low confidence and accuracy. To solve this problem, we apply our multi-view superquadric representation strategy to this scene.

Fig. 7.60 shows registered and integrated data of the scene from the four views, the part decomposition results, and the superquadric representation of the integrated data. This figure demonstrates that the four views are correctly registered using our view registration algorithm. The registered multi-view data is integrated using the RapidForm software. The integrated data



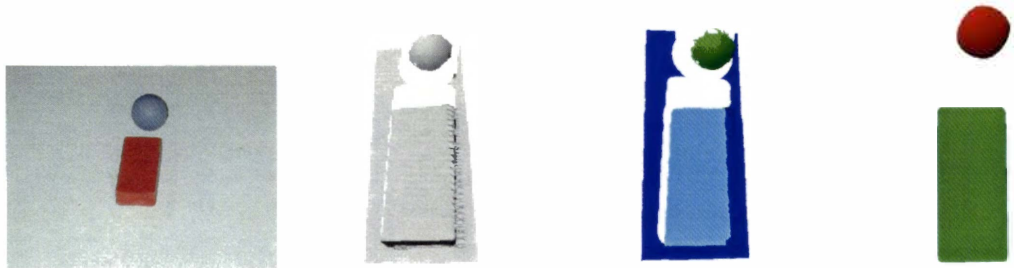
(a) (b) (c) (d)

Figure 7.57: Superquadric representation results for a scene (II) from view 2. The scene contains two separate objects including a block and a sphere. This mesh is a single-view scan from the IVP Ranger System. The mesh consists of 26,315 vertices and 50,804 triangles while the decomposition consists of three parts including the background. Each decomposed part is fitted to a superquadric model. Superquadrics are rendered in 3D using quad meshes based on recovered superquadric parameters. (a) Photograph of the original scene, (b) rendering of original mesh, (c) decomposed parts labeled in different colors, and (d) rendering of recovered superquadrics.



(a) (b) (c) (d)

Figure 7.58: Superquadric representation results for a scene (II) from view 3. The scene contains two separate objects including a block and a sphere. This mesh is a single-view scan from the IVP Ranger System. The mesh consists of 27,781 vertices and 53,654 triangles while the decomposition consists of three parts including the background. Each decomposed part is fitted to a superquadric model. Superquadrics are rendered in 3D using quad meshes based on recovered superquadric parameters. (a) Photograph of the original scene, (b) rendering of original mesh, (c) decomposed parts labeled in different colors, and (d) rendering of recovered superquadrics.

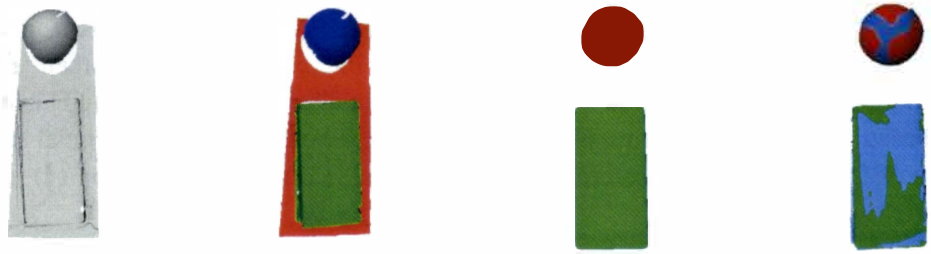


(a) (b) (c) (d)

Figure 7.59: Superquadric representation results for a scene (II) from view 4. The scene contains two separate objects including a block and a sphere. This mesh is a single-view scan from the IVP Ranger System. The mesh consists of 23,936 vertices and 46,374 triangles while the decomposition consists of three parts including the background. Each decomposed part is fitted to a superquadric model. Superquadrics are rendered in 3D using quad meshes based on recovered superquadric parameters. (a) Photograph of the original scene, (b) rendering of original mesh, (c) decomposed parts labeled in different colors, and (d) rendering of recovered superquadrics.

Table 7.12: Recovered superquadric parameters for a scene (II). The scene contains a block and a sphere. Four single views as well as integrated data are used.

Object	Data	a_1	a_2	a_3	ε_1	ε_2
Block	**Ground truth**	**14.0**	**28.0**	**58.5**	**0.1**	**0.1**
	View 1	11.71	27.85	57.59	0.1	0.17
	View 2	11.12	27.31	55.53	0.1	0.1
	View 3	14.41	27.5	57.9	0.1	0.1
	View 4	10.85	27.71	58.28	0.1	0.1
	Integrated data	**15.69**	**29.35**	**57.45**	**0.1**	**0.12**
Sphere	**Ground truth**	**29.5**	**29.5**	**29.5**	**1.0**	**1.0**
	View 1	9.83	20.25	20.48	1.11	1.13
	View 2	8.98	19.63	19.78	1.12	1.19
	View 3	9.45	20.12	20.28	1.14	1.19
	View 4	9.19	19.42	20.17	1.17	1.12
	Integrated data	**20.89**	**26.94**	**25.49**	**1.08**	**1.08**



(a) (b) (c) (d)

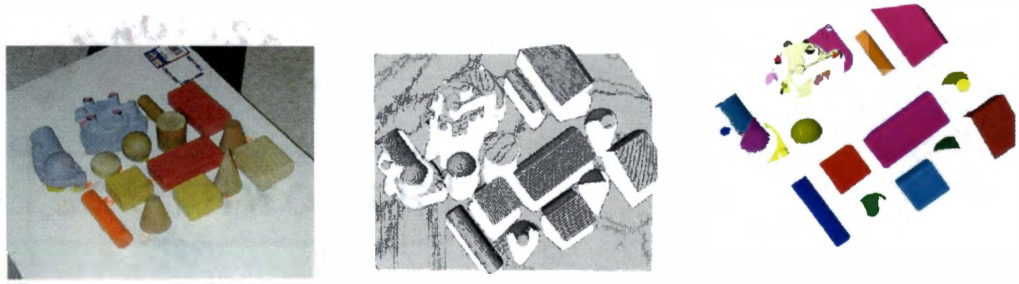
Figure 7.60: Multi-view superquadric representation results for a scene (II). The original scene is scanned from four different views. Superquadrics are recovered from each single view. The four-view data are registered based on the recovered superquadric parameters. The registered multi-view data are integrated using the RapidForm software. The integrated mesh consists of 30,118 vertices and 58,758 triangles while the decomposition consists of three parts including the background. The integrated data are then decomposed into single parts. Each decomposed part is fitted to a superquadric model. Superquadrics and original mesh for the scene are rendered in the same coordinate system to illustrate the fitting error. (a) Rendering of registered and integrated mesh of the original scene scanned from four views, (b) decomposed parts labeled in different colors, (c) rendering of recovered superquadrics, and (d) merged rendering of original mesh and recovered superquadrics in the same coordinate system.

(a) (b) (c) (d)

Figure 7.61: Superquadric representation results for a scene (III) from view 1. The scene contains miscellaneous objects simulating a bin picking application. This mesh is a single-view scan from view 1 using the IVP Ranger System. The mesh consists of 51,121 vertices and 96,401 triangles while the decomposition consists of 30 parts. Since this scan is from a single view, the mesh leaves the objects as isolated and thus unconnected to each other. Each decomposed part is fitted to a superquadric model. Superquadrics are rendered in 3D using quad meshes based on recovered superquadric parameters. (a) Photograph of the original scene, (b) rendering of original mesh, (c) decomposed parts labeled in different colors, and (d) rendering of recovered superquadrics.

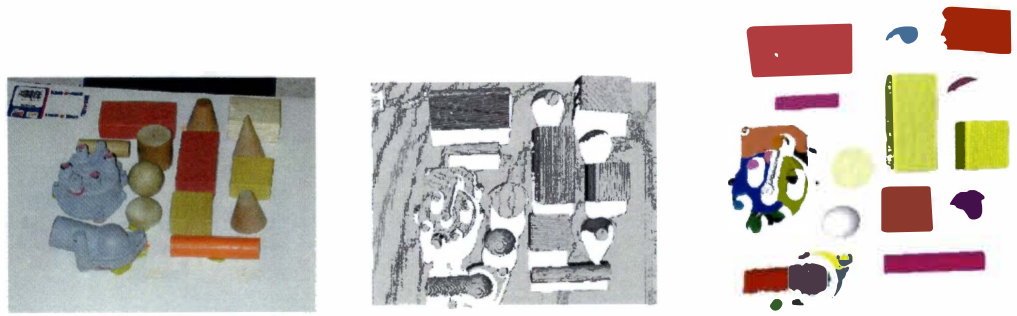
contains much more data than any single-view scan, especially for the sphere. In Fig. 7.60(d), superquadrics and original mesh for the scene are rendered in the same coordinate system to illustrate the fitting error between the raw data and the recovered superquadrics. This figure shows that recovered superquadrics correctly fit the original data with small fitting errors in the sense of visualization. As shown in Table 7.12, for both the block and the sphere, the superquadric parameters recovered from the integrated data are more accurate than from any of the four views. In addition, superquadrics recovered from the integrated data have higher confidence due to more complete data utilized. Consequently, we conclude that our multi-view superquadric representation algorithm is able to recover superquadrics with higher confidence and accuracy than single-view superquadric representation approaches.

Figs. 7.61 through 7.65 show part decomposition and superquadric representation results for a scene (III) from five views. This scene contains miscellaneous objects simulating a bin picking application. From the recovered superquadrics shown in Figs. 7.61(d) through 7.65(d), we observe that different superquadrics are recovered for the objects in the scene from the five views. This demonstrates the weakness caused by single-view superquadric representation, i.e., superquadrics recovered from single views are subject to viewpoints from which the scan is acquired. Therefore, superquadrics recovered from single views tend to be inaccurate due to partial data used. Without ground truths as priori knowledge about the scene, the confidence of superquadrics recovered from single views remain unknown. The experiments demonstrate that superquadrics recovered from single-view superquadric representation strategies have low confidence and accuracy. To address this problem, we apply our multi-view superquadric representation strategy to this scene.



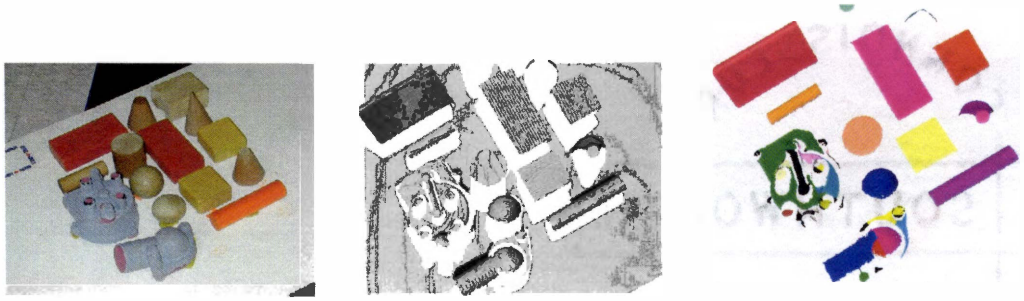
(a) (b) (c)

Figure 7.62: Superquadric representation results for a scene (III) from view 2. The scene contains miscellaneous objects simulating a bin picking application. This mesh is a single-view scan from view 2 using the IVP Ranger System. The mesh consists of 60,287 vertices and 114,768 triangles while the decomposition consists of 25 parts. Since this scan is from a single view, the mesh leaves the objects as isolated and thus unconnected to each other. Each decomposed part is fitted to a superquadric model. Superquadrics are rendered in 3D using quad meshes based on recovered superquadric parameters. (a) Photograph of the original scene, (b) rendering of original mesh, and (c) decomposed parts labeled in different colors.



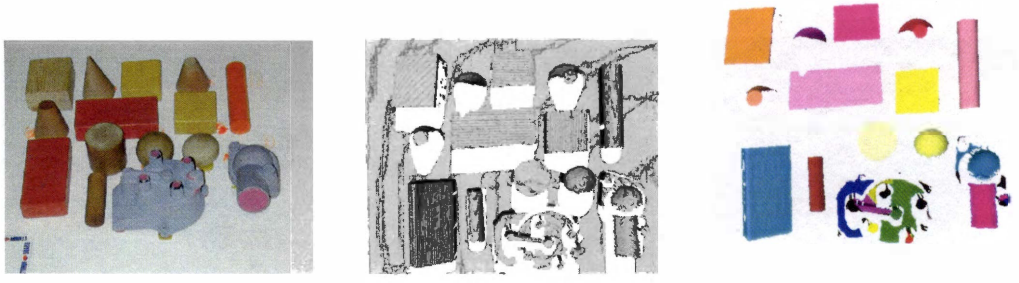
(a) (b) (c)

Figure 7.63: Superquadric representation results for a scene (III) from view 3. The scene contains miscellaneous objects simulating a bin picking application. This mesh is a single-view scan from view 3 using the IVP Ranger System. The mesh consists of 61,867 vertices and 118,016 triangles while the decomposition consists of 26 parts. Since this scan is from a single view, the mesh leaves the objects as isolated and thus unconnected to each other. Each decomposed part is fitted to a superquadric model. Superquadrics are rendered in 3D using quad meshes based on recovered superquadric parameters. (a) Photograph of the original scene, (b) rendering of original mesh, and (c) decomposed parts labeled in different colors.



(a) (b) (c)

Figure 7.64: Superquadric representation results for a scene (III) from view 4. The scene contains miscellaneous objects simulating a bin picking application. This mesh is a single-view scan from view 4 using the IVP Ranger System. The mesh consists of 55,753 vertices and 105,717 triangles while the decomposition consists of 27 parts. Since this scan is from a single view, the mesh leaves the objects as isolated and thus unconnected to each other. Each decomposed part is fitted to a superquadric model. Superquadrics are rendered in 3D using quad meshes based on recovered superquadric parameters. (a) Photograph of the original scene, (b) rendering of original mesh, and (c) decomposed parts labeled in different colors.



(a) (b) (c)

Figure 7.65: Superquadric representation results for a scene (III) from view 5. The scene contains miscellaneous objects simulating a bin picking application. This mesh is a single-view scan from view 5 using the IVP Ranger System. The mesh consists of 62,091 vertices and 117,934 triangles while the decomposition consists of 24 parts. Since this scan is from a single view, the mesh leaves the objects as isolated and thus unconnected to each other. Each decomposed part is fitted to a superquadric model. Superquadrics are rendered in 3D using quad meshes based on recovered superquadric parameters. (a) Photograph of the original scene, (b) rendering of original mesh, and (c) decomposed parts labeled in different colors.

Fig. 7.66 shows registered and integrated data of the scene from the five views, the part decomposition results, and the superquadric representation of the integrated data. This figure shows that the five views are correctly registered using our view registration algorithm. The registered data is integrated using the RapidForm software. For each object, the integrated data contains more information than any of the five single-view scans, especially for the water neck and the cones. Consequently, as shown in Fig. 7.66(c), superquadrics recovered from the integrated data are much better than those from each of the five views in terms of accuracy and confidence. Similar to the scenes shown previously, this set of experiments demonstrates that our multi-view superquadric representation algorithm is able to recover superquadrics with higher confidence and accuracy than single-view superquadric representation approaches.

Figs. 7.67 through 7.70 show part decomposition and superquadric representation results for a scene (V) from four views. This scene contains miscellaneous objects with significant occlusions simulating a bin picking application. From the recovered superquadrics shown in Figs. 7.67(d) through 7.70(d), we observe that different superquadrics are recovered for the objects in this scene from the four single views. This demonstrates the weakness caused by single-view superquadric representation, i.e., superquadrics recovered from single views are subject to viewpoints from which the scan is acquired. Therefore, superquadrics recovered from single views tend to be inaccurate due to partial data used. Without ground truths as priori knowledge about the scene, the confidence of superquadrics recovered from single views remain unknown. This set of experiments demonstrates that superquadrics recovered from single-view superquadric representation strategies have low confidence and accuracy. To solve this problem, we apply our multi-view superquadric representation strategy to the scene.

Fig. 7.71 shows registered and integrated data of the scene from the four views, the part decomposition results, and the superquadric representation of the integrated data. This figure illustrates that the four views are correctly registered using our view registration method. The registered data is integrated using the RapidForm software. For each object in the scene, the integrated data contains more information than a single-view scan, especially for the water neck and the mug. Consequently, as shown in Fig. 7.71(c), superquadrics recovered from the integrated data are much better than those from any of the five views in terms of accuracy and confidence. Similar to the scenes shown previously, this set of experiments demonstrates that our multi-view superquadric representation algorithm is able to recover superquadrics with higher confidence and accuracy than single-view superquadric representation approaches.

7.3.3 Comparison with the Recover-and-Select Algorithm

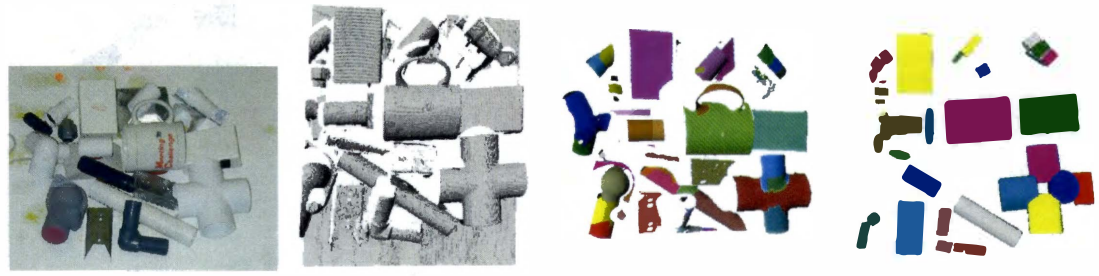
To demonstrate advantages of our multi-view superquadric representation algorithm to other single-view superquadric representation approaches, we compare our and the recover-and-select algorithms on a scene (V). Fig. 7.72 shows superquadric representation results for the scene using the recover-and-select and our multi-view superquadric representation algorithms. For the recover-and-select algorithm, we use the single-view range image acquired from view 1 as an example. For our method, the mesh is integrated from four views, and consists of 131,044 vertices and 228,726 triangles while the decomposition consists of 56 parts. Each of the decomposed parts is fitted to a superquadric model. As shown in Fig. 7.72(c), the recover-and-select algorithm fails to recover superquadrics for the crossing-shaped pipe fitting, the handle of the

(a) (b) (c)

Figure 7.66: Multi-view superquadric representation results for a scene (III). The original scene is scanned from five different views. Superquadrics are recovered from each single view. The five-view data are registered based on the recovered superquadric parameters. The registered multi-view data are integrated using the RapidForm software. The integrated mesh consists of 102,647 vertices and 189,007 triangles while the decomposition results consist of 29 parts. The integrated data are then decomposed into single parts. Each decomposed part is fitted to a superquadric model. Superquadrics are rendered in 3D using quad meshes based on recovered superquadric parameters. (a) Rendering of registered and integrated mesh of the original scene scanned from five views, (b) decomposed parts labeled in different colors, and (c) rendering of recovered superquadrics.

(a) (b) (c) (d)

Figure 7.67: Superquadric representation results for a scene (V) from view 1. The scene contains miscellaneous objects simulating a bin picking application. This mesh is a single-view scan from view 1 using the IVP Ranger System. The mesh consists of 89,561 vertices and 169,479 triangles while the decomposition consists of 49 parts. Since this scan is from a single view, the mesh leaves the objects as isolated and thus unconnected to each other. Each decomposed part is fitted to a superquadric model. Superquadrics are rendered in 3D using quad meshes based on recovered superquadric parameters. (a) Photograph of the original scene, (b) rendering of original mesh, (c) decomposed parts labeled in different colors, and (d) rendering of recovered superquadrics.

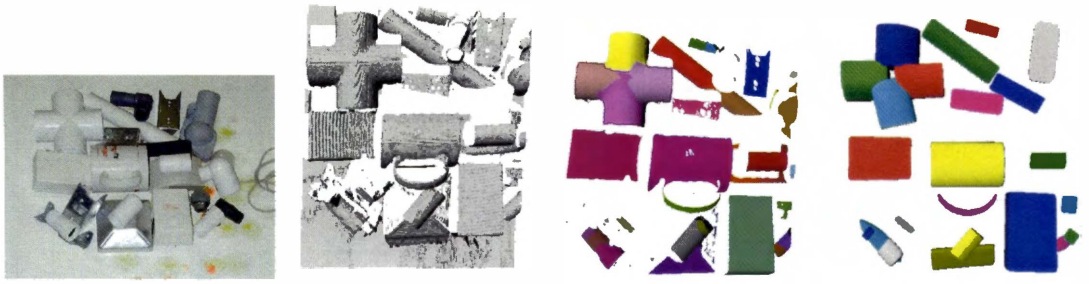


(a) (b) (c) (d)

Figure 7.68: Superquadric representation results for a scene (V) from view 2. The scene contains miscellaneous objects simulating a bin picking application. This mesh is a single-view scan from view 2 using the IVP Ranger System. The mesh consists of 95,855 vertices and 182,779 triangles while the decomposition consists of 54 parts. Since this scan is from a single view, the mesh leaves the objects as isolated and thus unconnected to each other. Each decomposed part is fitted to a superquadric model. Superquadrics are rendered in 3D using quad meshes based on recovered superquadric parameters. (a) Photograph of the original scene, (b) rendering of original mesh, (c) decomposed parts labeled in different colors, and (d) rendering of recovered superquadrics.

(a) (b) (c) (d)

Figure 7.69: Superquadric representation results for a scene (V) from view 3. The scene contains miscellaneous objects simulating a bin picking application. This mesh is a single-view scan from view 3 using the IVP Ranger System. The mesh consists of 101,502 vertices and 194,558 triangles while the decomposition consists of 35 parts. Since this scan is from a single view, the mesh leaves the objects as isolated and thus unconnected to each other. Each decomposed part is fitted to a superquadric model. Superquadrics are rendered in 3D using quad meshes based on recovered superquadric parameters. (a) Photograph of the original scene, (b) rendering of original mesh, (c) decomposed parts labeled in different colors, and (d) rendering of recovered superquadrics.



(a) (b) (c) (d)

Figure 7.70: Superquadric representation results for a scene (V) from view 4. The scene contains miscellaneous objects simulating a bin picking application. This mesh is a single-view scan from view 4 using the IVP Ranger System. The mesh consists of 97,235 vertices and 185,464 triangles while the decomposition consists of 36 parts. Since this scan is from a single view, the mesh leaves the objects as isolated and thus unconnected to each other. Each decomposed part is fitted to a superquadric model. Superquadrics are rendered in 3D using quad meshes based on recovered superquadric parameters. (a) Photograph of the original scene, (b) rendering of original mesh, (c) decomposed parts labeled in different colors, and (d) rendering of recovered superquadrics.

(a) (b) (c)

Figure 7.71: Multi-view superquadric representation results for a scene (V). The original scene is scanned from four different views. Superquadrics are recovered from each single view. The four-view data are registered based on the recovered superquadric parameters. The registered multi-view data are integrated using the RapidForm software. The integrated mesh consists of 131,044 vertices and 228,726 triangles while the decomposition consists of 56 parts. The integrated data are then decomposed into single parts. Each decomposed part is fitted to a superquadric model. Superquadrics are rendered in 3D using quad meshes based on recovered superquadric parameters. (a) Rendering of registered and integrated mesh of the original scene scanned from four views, (b) decomposed parts labeled in different colors, and (c) rendering of recovered superquadrics.

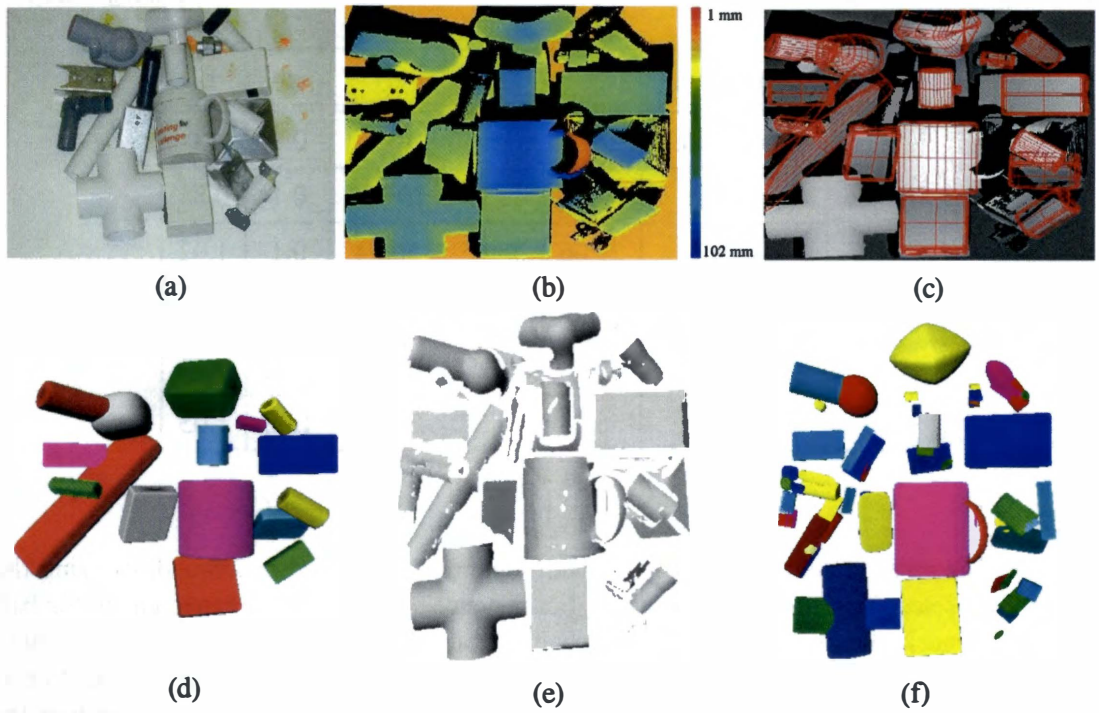


Figure 7.72: Comparison of superquadric representation results for a scene (V). The scene contains miscellaneous objects simulating a bin picking application. Superquadrics are recovered for the scene using two algorithms including our superquadric representation strategy utilizing part decomposition and the recover-and-select algorithm. The original range image is pseudo-colored to illustrate the range depth variations on the surfaces. Red corresponds to larger distances between the sensor and the object surface while blue corresponds to smaller distances. For the recover-and-select algorithm, the single-view range image is acquired from view 1, and no pre-segmentation is involved. For our method, the mesh is integrated from four views, and consists of 131,044 vertices and 228, 726 triangles while the decomposition consists of 56 parts. Each of the decomposed parts is fitted to a superquadric model. Superquadrics are rendered in 3D using quad meshes based on recovered superquadric parameters. (a) Photograph of original scene, (b) pseudo-colored range image of original scene, (c) recovered superquadrics using the recover-and-select algorithm and plotted by red wireframes, (d) rendering of superquadrics recovered from the recover-and-select algorithm, (e) rendering of original mesh, and (f) rendering of recovered superquadrics using our superquadric representation algorithm.

Table 7.13: Comparison of recovered superquadric parameters for a scene (V). The recover-and-select and our multi-view superquadric representation algorithms are used. Superquadric parameters are recovered and compared for the handle and the ball of the water neck. RAS: the recover-and-select algorithm. MVSR: our multi-view superquadric representation algorithm.

Object	Algorithm	a_1	a_2	a_3	ε_1	ε_2
Handle	Ground truth	39.7	39.4	67.6	0.1	1.0
	RAS	33.82	49.24	104.84	0.1	0.84
	MVSR	22.06	38.85	56.94	0.1	1.02
Ball	Ground truth	50.0	47.6	56.0	1.0	1.0
	RAS	64.0	91.5	105.66	0.72	0.58
	MVSR	50.18	45.37	37.62	1.02	1.05

mug, and the small bolts in the scene. In addition, some recovered superquadrics using the recover-and-select algorithm are inaccurate. For instance, superquadrics recovered for the ball of the water neck, the long pipe, and the mechanical part are inaccurate compared with ground truths. Fig. 7.72(f) demonstrates that our multi-view superquadric representation algorithm is able to recover more accurate superquadrics for these objects. In particular, superquadrics for the water neck, the crossing-shaped pipe fitting, the mug, the power adapters, etc. are successfully recovered using our algorithm. The recovered superquadric parameters for the handle and the ball of the water neck using the two algorithms are illustrated in Table 7.13. We observe that our algorithm recovers much more accurate parameters for these objects than the recover-and-select algorithm when compared with ground truths.

7.3.4 Remarks

Based on the experimental results demonstrated in this section, the following conclusions on the proposed multi-view superquadric representation algorithm are derived.

- The proposed multi-view superquadric representation algorithm significantly improves the confidence and accuracy of recovered superquadrics by incorporating multi-view data. The number of views needed depends on the complexity of the scene, and generally ranges from two to ten. Experimental results demonstrate that the proposed multi-view superquadric representation algorithm is able to improve the accuracy of recovered superquadric parameters up to 75 times over the single-view recovery results.

- The algorithm is able to represent multi-part objects and multi-object scenes with occlusions by first decomposing them into single parts. We use calibrated 3D range data

(triangle meshes) instead of uncalibrated range images as input, which enables recovered superquadrics to represent the true geometry of the objects. In addition, the algorithm is able to recover both regular and globally deformed superquadrics.

- The performance comparison between our multi-view superquadric representation and the recover-and-select algorithms demonstrates that our algorithm can recover more complete and more accurate superquadrics. Compared with the recover-and-select algorithm, our algorithm is more flexible and efficient.

7.4 Summary

This chapter presented extensive experimental results for the contributions proposed in this dissertation. A wide variety of realistic objects and scenes were used in the experiments. To name a few, the objects we have used include a mug, a computer mouse, two PVC pipe fittings, automotive parts, mechanical parts, etc. For each set of experimental results, we have provided both qualitative and quantitative analysis. In addition, we demonstrated advantages of our proposed algorithms by comparing the performance of our algorithms with existing approaches through experiments. The experiments demonstrate that our proposed algorithms extend the state of the art of superquadric representation in the following aspects. (i) Multi-part objects and multi-object scenes can be successfully represented by superquadrics using our method, (ii) compared with range images, superquadrics recovered from 3D calibrated range data (triangle meshes) represent the true geometry of the objects, and (iii) superquadrics recovered using our algorithm have higher confidence and accuracy.

Chapter 8

Conclusions

In this dissertation, we have proposed new strategies to address superquadric representation and related problems in computer vision. In particular, we propose a multi-view superquadric representation algorithm that is able to represent multi-part objects and multi-object scenes with superquadrics of high confidence and accuracy. Within this framework, a new 3D part decomposition algorithm based on curvature analysis is presented. In addition, we present an improved and extended view registration algorithm based on superquadrics. We summarize the proposed contributions in Section 8.1 and the advantages of superquadric representation in Section 8.2. In Section 8.3, we present a short discussion of future directions to this work. Finally, Section 8.4 concludes this dissertation.

8.1 Summary of Contributions

This dissertation presents three major contributions in superquadric representation, part decomposition, and view registration. The proposed algorithms extend the state of the art in these three fields. In particular, we have proposed a **multi-view superquadric representation algorithm** of which the advantage is representing multi-part objects or multi-object scenes with superquadrics in high fidelity and accuracy. Within the multi-view superquadric representation framework, we have presented a **3D part decomposition algorithm** based on transversality regularity to automatically decompose multi-part objects into their constituent single parts. Finally, we have extended a **view registration algorithm** based on superquadrics to register multi-view 3D range data in a pairwise manner.

Due to occlusions inherent in single-view range images, a single-view scan can only acquire partial information from a scene. Therefore, superquadrics recovered from single-view images have low confidence. To address this problem, our proposed multi-view superquadric representation algorithm utilizes multi-view 3D range data sets. In order to represent complex objects and scenes, we present a 3D part decomposition algorithm as a pre-segmentation step. We also present an improved and extended 3D view registration algorithm that utilizes recovered superquadrics. Superquadrics are then recovered from registered and integrated data and therefore have higher confidence. The three contributions proposed in this dissertation are summarized as follows.

Multi-view superquadric representation. The most significant contribution is the development of a multi-view superquadric representation algorithm. The multi-view superquadric representation algorithm addresses two unsolved problems in the literature. The first problem is superquadric representation of multi-part objects or multi-object scenes. The second one is the low confidence and accuracy of superquadrics recovered from single-view range images. Correspondingly, our proposed multi-view superquadric representation algorithm has two main advantages compared to existing superquadric representation methods in the literature. First, our algorithm is able to efficiently represent multi-part and multi-object scenes with superquadrics. The incorporation of a part decomposition step turns the difficult superquadric representation of multi-part objects or multi-object scenes into a straightforward data fitting problem. Second, our proposed superquadric representation algorithm is able to achieve high confidence and accuracy of recovered superquadrics by utilizing multi-view range data. The diagram of the proposed multi-view superquadric representation algorithm is illustrated in Fig. 8.1. This algorithm is presented in Chapter 3.

3D part decomposition. The second major contribution is the development of a 3D part decomposition algorithm within the proposed multi-view superquadric representation framework as shown in Fig. 1.8. The proposed part decomposition algorithm is based on the principal of transversality regularity (Hoffman and Richards, 1984). Different from surface or region segmentation, the proposed part decomposition algorithm segments multi-part objects into their constitute parts that are recognizable to human perception systems. This algorithm takes surfaces represented by triangular meshes as input. Compared to the part decomposition algorithm (Wu and Levine, 1997) using simulated electrical charge distributions, the proposed algorithm can be easily implemented and has reliable performances. The most significant advantage of the proposed part decomposition algorithm is that it directly facilitates the superquadric representation of multi-part objects or multi-object scenes. The proposed part decomposition algorithm is presented in Chapter 4.

View registration based on superquadrics. The final contribution is a 3D view registration algorithm based on superquadrics. This algorithm extends the range image registration algorithm (Jaklič et al., 2000) in three aspects. First, we have derived the first- and second-order inertial moments for tapered superquadrics, which enable tapered superquadrics to be registered with the extended algorithm. Second, the original 2D range image registration algorithm is extended in our algorithm to register 3D surfaces. Finally, the extended and improved view registration algorithm can register scenes consisting of complex background and objects under the assumption that at least one object in the scene can be represented by a superquadric and that the object is visible (scannable) from views to be registered. Compared with correspondence-based view registration methods, our algorithm requires less overlap between views and is less time-consuming. As view registration is indispensable to computer vision tasks involving multi-view data, the proposed view registration algorithm is crucial to our multi-view superquadric representation algorithm. The proposed view registration algorithm is presented in Chapter 5.

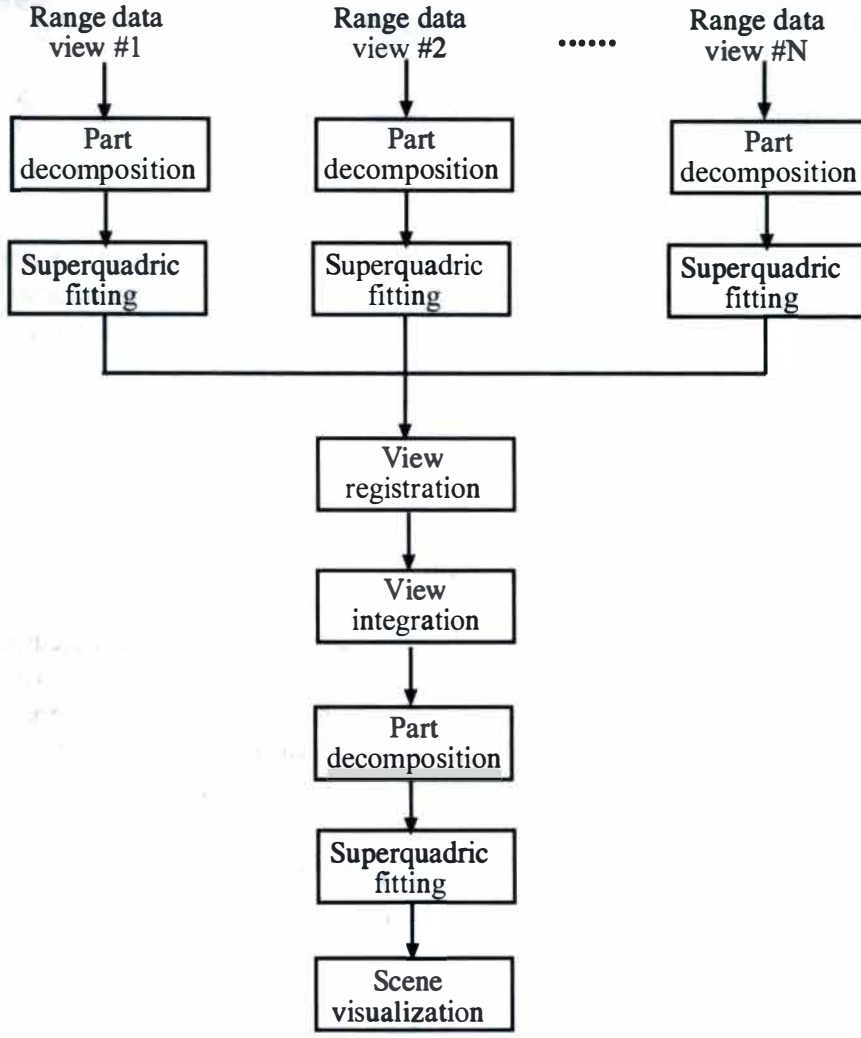
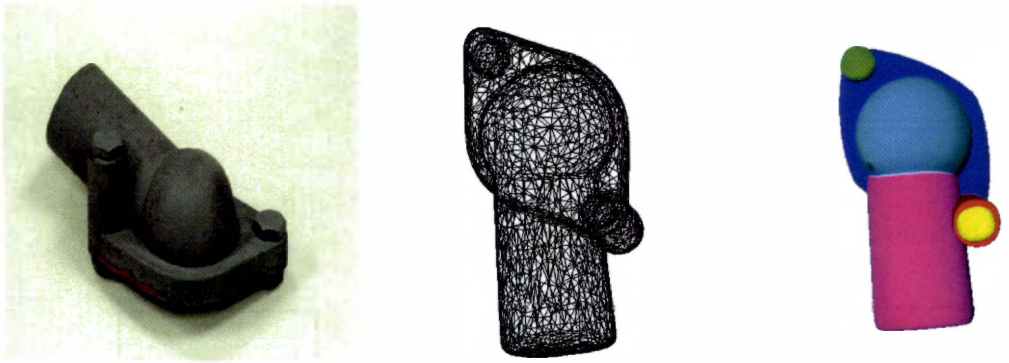


Figure 8.1: Diagram of the proposed multi-view superquadric representation algorithm. Triangle 3D meshes for the range images are used as input. Part decomposition is utilized for multi-part objects or multi-object scene. The number of views depends on the complexity of the objects or scenes and the tasks involved. View integration is performed by RapidForm software.



(a) (b) (c)

Figure 8.2: Superquadric representation results for a water neck. This mesh is a reconstruction from multiple range images using the IVP Ranger System. The mesh consists of 58,784 vertices and 117,564 triangles. Each decomposed part is fitted to a superquadric model. Each superquadric is rendered in 3D using quad meshes based on recovered superquadric parameters. (a) Rendering of original mesh, (b) wireframes of triangle meshes, and (c) rendering of recovered superquadrics.

8.2 Advantages of Superquadrics

Superquadrics are not capable of representing complicated details of objects accurately due to the models' parametric characteristics. However, the advantage of superquadrics is that they can provide quantitative and compact information about objects including size, position, and orientation in the world coordinate system. The quantitative information provided by the represented superquadrics could significantly benefit object recognition, object manipulation, etc. Moreover, superquadric representation provides a much higher data compression ratio than polygonal meshes. The data compression ratio DCR is defined in Chapter 7 as

$$DCR = \frac{3 \times N(V) + 3 \times N(T)}{m \times N(SQ)}. \tag{8.1}$$

For a multi-part object such as a water neck shown in Fig. 8.2, the reconstructed mesh consists of 58,784 vertices and 117,564 triangles. For superquadric representation, nine superquadrics are recovered with a total of only 99 parameters representing the water neck. As shown in Fig. 8.2(c), superquadric representation achieves a high visualization quality with only a small number of parameters (99 vs. over 100,000 numbers). The data compression ratio DCR is evaluated as 5343.88. Such a huge data reduction and the significantly lower storage cost of superquadric representation enables tasks such as visualization, transmission, manipulation, CAD modeling, etc. to be performed in real time, which is a huge advantage over mesh representation.

Furthermore, we explore the advantages of superquadric representation over mesh reduction. Fig. 8.3 shows a comparison of mesh reduction and superquadric representation for a

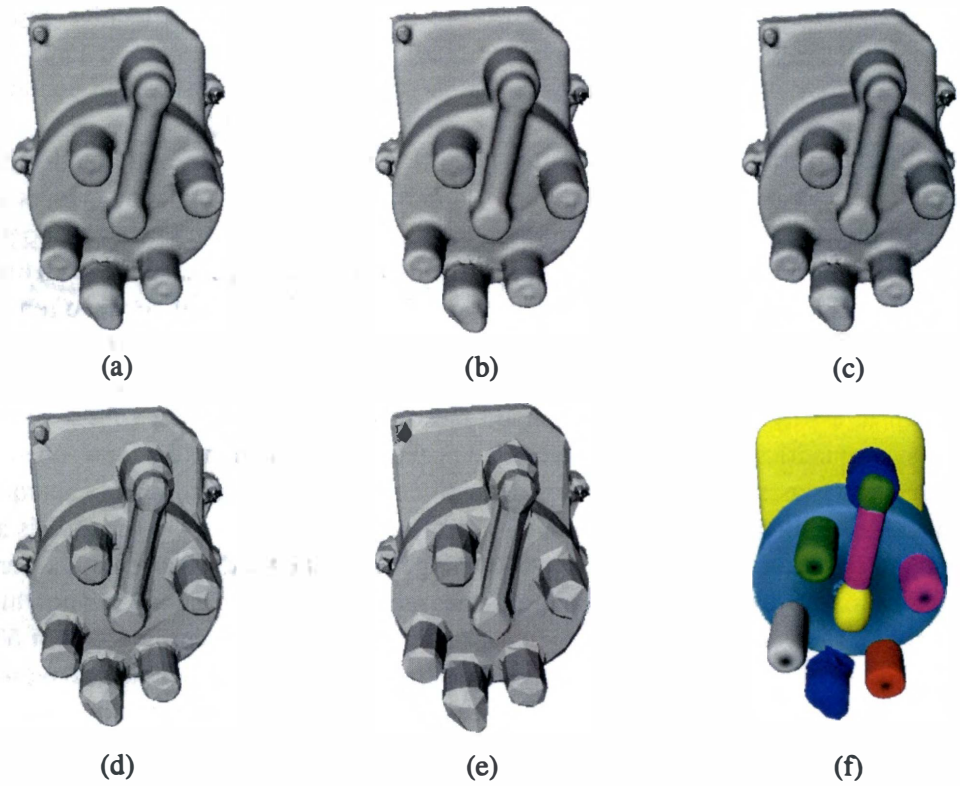


Figure 8.3: Mesh reduction and superquadric representation for a distributor cap. This mesh is a reconstruction from multiple range images using the IVP Ranger System. The original mesh consists of 58,975 vertices and 117,036 triangles. The original mesh is reduced iteratively to four levels by the mesh decimation function (high quality) in the RapidForm software. The original mesh is also decomposed and represented by 13 superquadrics from our algorithm. (a) Rendering of original mesh, (b) rendering of reduced mesh consisting of 29,681 vertices and 58,512 triangles, (c) rendering of reduced mesh consisting of 15,021 vertices and 29,254 triangles, (d) rendering of reduced mesh consisting of 1,767 vertices and 2,925 triangles, (e) rendering of reduced mesh consisting of 971 vertices and 1,455 triangles, and (f) rendering of recovered superquadrics.

distributor cap. The original mesh shown in Fig. 8.3(a) is reconstructed from multi-view range scans from the the IVP Ranger System (Integrated Vision Products, 2000), and consists of 58,975 vertices and 117,036 triangles. We iteratively reduce the number of triangles using the mesh decimation function (high quality) in the RapidForm software (INUS Technology, Inc, 2001) to four levels as the first reduced mesh consists of 29,681 vertices and 58,512 triangles; the second one consists of 15,021 vertices and 29,254 triangles; the third one consists of 1,767 vertices and 2,925 triangles; the last one consists of 971 vertices and 1,455 triangles. Fig. 8.3(b) and Fig. 8.3(c) show that little visual difference can be observed between the first and the second reduced meshes and the original mesh since these two reduced meshes still contain huge numbers of vertices and triangles.

However, Fig. 8.3(d) and Fig. 8.3(e) show that rendered surfaces become rough when the numbers of triangles are reduced to 2,925 and 1,455 respectively. This phenomenon is a common drawback of mesh reduction, i.e., mesh reduction reduces the visual surface smoothness, especially when the mesh reduction ratio is high (the number of triangles is reduced 40 times for the mesh shown Fig. 8.3(d) and 80 times for the mesh shown Fig. 8.3(e)). How to represent a 3D surface model with the least number of vertices and triangles while maintaining the acceptable surface smoothness is an ongoing research area. Superquadric representation addresses this problem in a way different from common mesh reduction approaches. As a volumetric primitive, superquadrics can represent objects compactly, i.e., 11 parameters for a superquadric can represent thousands of triangle meshes. In addition, the explicit definition of superquadrics enables them to be rendered smoothly in 3D. Therefore, superquadric representation is a natural solution to the difficulties in mesh reduction. Fig. 8.3(f) shows 13 recovered and rendered superquadrics for the distributor cap. Superquadrics represent the distributor cap smoothly with only a total of 143 floating numbers. Superquadric representation achieves a DCR of 50.9 for the reduced mesh shown in Fig. 8.3(e) and demonstrates a much smoother surface representation as shown in Fig. 8.3(f). Such characteristics enable superquadrics to significantly benefit many tasks such as scene visualization, object recognition, data transmission, etc.

8.3 Future Research

Several extensions to this work are possible. Since this research concentrates on object representation, it is straightforward to extend this research to object recognition, in particular, 3D part-based object recognition. Object recognition is crucial in robot navigation tasks. How color information can be utilized to facilitate object recognition is also a meaningful topic.

Due to the parametric feature of superquadrics, shapes that can be represented by superquadrics are still limited even with global and free-form deformations. In contrast, superquadrics with local deformations are able to represent arbitrary shapes with high computation and storage costs. The recovery of superquadrics with local deformations needs to be explored.

Another possible extension is the improvement of the proposed part decomposition algorithm. A more accurate estimation of Gaussian curvature would improve the performance of the part decomposition algorithm. In addition, a faster neighborhood searching algorithm for

triangle meshes would significantly improve the segmentation speed. Part decomposition based on other theories such as the minima rule could also be a possible extension to this work.

8.4 Summary

In this dissertation, we have described the motivations for the contributions we present, and we have reviewed the related literature. We then presented three algorithms as our major contributions along with experimental results to support the algorithms. For each proposed algorithm, advantages and limitations were summarized. With the development of our algorithms, we have achieved great flexibilities of representing complex objects and scenes with superquadrics. Finally, future directions to this research were discussed.

Bibliography

Bibliography

Adams, M. and Robert, P. (1996). The interpretation of phase and intensity data from amcw light detection and ranging sensors. *International Journal of Robotics Research*, 15(5):441–458.

Astrand, E., Johannesson, M., and Astrŏm, A. (1995). Two novel methods for range imaging using the mapp2200 smart sensor. In *Proc. SPIE Conf. Videometrics II*, pages 170–181.

Astrŏm, A. (1993). *Smart image sensors.* Ph.D. thesis, Department of Electrical Engineering, Linkŏ ping University, Sweden.

Baccar, M., Gee, L., Gonzalez, R., and Abidi, M. (1996). Segmentation of range images via data fusion and morphological watersheds. *Pattern Recognition*, 29(10):1671–1685.

Bajcsy, R. and Solina, F. (1987). Three-dimensional object representation revisited. In *Int'l Conf. Computer Vision.*

Bajcsy, R., Solina, F., and Gupta, A. (1990). Segmentation versus object representation, are they separable? In *Analysis and interpretation of range images*, pages 207–217. Springer-Verlag.

Bardinet, E., Cohen, L., and Ayache, N. (1998). A parametric deformable model to fit unstructed 3D data. *Computer Vision and Image Understanding*, 71(1):39–54.

Barr, A. (1981). Superquadrics and angle-preserving transformations. *IEEE Trans. Computer Graphics and Applications*, 1(1):11–23.

Barr, A. (1984). Global and local deformations of solid primitives. *Computer Graphics*, 18(3):21–30.

Bennamoun, M. (1994). A contour-based part segmentation algorithm. In *Int'l Conf. Acoustics, Speech, and Signal Processing*, volume 5, pages 41–44.

Beraldin, J., El-Hakim, S., and Cournoyer, L. (1993). Practical range camera calibration. In *Proc. SPIE Conf. Videometrics II*, pages 21–31.

Bergevin, R., Laurendeau, D., and Poussart, D. (1995). Registering range views of multipart objects. *Computer Vision and Image Understanding*, 61(1):1–16.

Bergevin, R., Soucy, M., Gagnon, H., and Laurendeau, D. (1996). Towards a general multi-view regsitration technique. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 18:540–547.

Besl, P. (1988). *Surfaces in range image understanding*. Springer-Verlag Press, Berlin.

Besl, P. (1989). Active optical range imaging sensors. In *Advances in machine vision*, chapter 1, pages 1–63. Springer-Verlag, U.S.

Besl, P. (1990). The free-form surface matching problem. In *Machine vision for three-dimensional scenes*, pages 25–71. Academic Press, San Diego.

Besl, P. and McKay, N. (1992). A method for registration of 3D shapes. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 14(2):239–256.

Best, L. and Magee, M. (1998). Autonomous construction of three-dimensional models from range data. *Pattern Recognition*, 31(2):121–136.

Biederman, I. (1985). Human image understanding: recent research and theory. *Computer vision, graphics, and image processing*, 32:29–733.

Binford, T. O. (1971). Visual perception by computer. In *IEEE Conf. On Systems and Controls.*

Blais, G. and Levine, M. D. (1995). Registering multiview range data to create 3D computer objects. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 16(10):820–824.

Boughorbel, F., Zhang, Y., Kang, S., and et al (2003). Laser ranging and video imaging for bin picking. *Assembly Automation*, 23(1):53–59.

Brady, M. (1985). Describing surfaces. *IEEE Trans. Computer vision, graphics, and image processing*, 32: 1–28.

Brooks, R. (1983). Model-based 3D interpretation of 2D images. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 5(2):140–150.

Burgiss, S., Whitaker, R., and Abidi, M. (1998). Range image segmentation through pattern analysis of the multiscale wavelet transform. *digital Signal Processing*, 8:267–276.

Campbell, R. and Flynn, P. (2001). A survey of free-form object representation and recognition techniques. *Computer Vision and Image Understanding*, 81:166–210.

Caselles, V., Kimmel, R., and Sapiro, G. (1997). Geodesic active contours. *International Journal of Computer Vision*, 22(1):61–79.

Castro, G., Nieto, J., Gallego, L., Pastor, L., and Cabello, E. (1998). An effective camera calibration method. In *Int'l Conf. Advanced Motion Control*, pages 171–174.

Center for Vision, Speech, and Signal Processing (2002). http://www.ee.surrey.ac.uk/research/vssp/demos/css3d.

Chan, M. and Metaxas, D. (1994). Physics-based object pose and shape estimation from multiple views. In *Int'l Conf. Pattern Recognition*, pages 326–330.

Chen, C., Huang, T., and Arrott, M. (1994). Modeling, analysis, and visualization of left ventricle shape and motion by hierarchical decomposition. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 16(4):342–356.

Chen, Y. and Medioni, G. (1992). Object modeling by registration of multiple range images. *Image and Vision Computing*, 10(3):145–155.

Chen, Y. and Medioni, G. (1994). Fitting a surface to 3D points using an inflating balloon model. In *2nd CAD-based Vision Workshop*, volume 13, pages 266–273.

Chua, C. and Jarvis, R. (1996). 3-D free-form surface registration and object recognition. *International Journal of Computer Vision*, 17(1):99–99.

Chua, C. and Jarvis, R. (1997). Point signatures: a new representation for 3D object recognition. *International Journal of Computer Vision*, 25(1):63–85.

Cohen, I. and Cohen, L. (1996). A hybrid hyperquadric model for 2D and 3D data fitting. *Computer Vision and Image Understanding*, 63(3):527–541.

Cohen, L., Bardinet, E., and Ayache, N. (1993). Surface Reconstruction Using Active Contour Models. Technical Report 1824, NYIT Computer Graphics Lab.

Collier, R. (1998). Characterization of a range scanning system utilizing a point laser rangefinder. Master's Thesis, University of Tennessee, Knoxville.

Computational geometry algorithms library (2002). http://www.cgal.org.

Connolly, C. (1999). The determination of next best views. In *Int'l Conf. Robotics and Automation*, volume 15, pages 57–66.

Curless, B. and Levoy, M. (1996). A volumetric method for building complex models from range images. *Proc. Computer Graphics (SIGGRAPH)*, pages 303–312.

D. Nitzan, A. B. and Duda, R. (1977). The measurement and use of registered reflectance data in scene analysis. In *IEEE Proceedings*, pages 206–220.

Delingette, H. (1997). General object reconstruction based on simplex meshes. Technical report, INRIA, France.

Desbrun, M., Meyer, M., Schröder, P., and Barr, A. (1999). Implicit fairing of irregular meshes using diffusion and curvature flow. *Proc. Computer Graphics (SIGGRAPH)*, pages 317–324.

Dickinson, S., Christensen, H., Tsotos, J., and Olofsson, G. (1994). Active Object Recognition Integrating Attention and Viewpoint Control. Technical report, Center for Cognitive Science, Rutgers University.

Dickinson, S. and Metaxas, D. (1992). Integrating qualitative and quantitative object representations in the recovery and tracking of 3D shapes. Technical report, Center for Cognitive Science, Rutgers University.

Dickinson, S., Metaxas, D., and Pentland, A. (1997). The role of model-based segmentation in the recovery of volumetric parts from range data. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 19(3):259–267.

Dickinson, S. and Pentland, A. (1992). 3D shape recovery using distributed aspect matching. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 14(2):174–198.

Dickinson, S., Pentland, A., and Rosenfeld, A. (1992). From volumes to views: an approach to 3D object recognition. *CVGIP: Image Understanding*, 55(2):130–154.

Dion, D., Laurendeau, D., and Bergevin, R. (1997). Telerobotic scene description using generalized cylinders. In *Proc. Vision Interface*, pages 81–89.

do Carmo, M. (1976). *Differential geometry of curves and surfaces.* Englewood Cliffs, NJ: Prentice-Hall, U.S.

Dop, E. V. (1999). *Multi-sensor object recognition: the case of electronics recycling*. Ph.D. thesis, University of Twente.

Dop, E. V. and Regitien, P. (1998). Fitting undeformed superquadrics to range data:improving model recovery and classification. In *Int'l Conf. Computer Vision and Pattern Recognition*, volume 2, pages 396–401.

Dorai, C. and Jain, A. (1997). Cosmos-a representation scheme for 3D free-form objects. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 19(10):1115–1129.

Dorai, C., Wang, G., Jain, A., and Mercer, C. (1996). From images to models: automatic 3D object model construction from multiple views. In *Int'l Conf. Pattern Recognition*, pages 770–774.

Dorai, C., Wang, J., Jain, A. K., and Mercer, C. (1997). Optimal registration of object views using range data. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 19:1131–1138.

Dorai, C., Wang, J., Jain, A. K., and Mercer, C. (1998). Registration and integration of multiple object views for 3D model construction. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 20:83–89.

Eggart, D., Fitzgibbon, A., and Fisher, R. (1996). Simultaneous registration of multiple range views for reverse engineering. In *Int'l Conf. Pattern Recognition*, pages 243–247.

Falcidieno, B. and Spagnuolo, M. (1992). Polyhderal surface decomposition based on curvature analysis. In *Model geometric computing for visualization*, pages 57–72. Springer-Verlag.

Faugeras, O. (1993). *Three-dimensional computer vision.* MIT Press, Cambridge, MA.

Feldmar, J. and Ayache, N. (1996). Rigid, affine, and locally affine registration of free-form surfaces. *International Journal of Computer Vision*, 18(2):99–119.

Ferrie, F., Lagarde, J., and Whaite, P. (1993). Darboux frames, snakes, and super-quadrics: geometry from the bottom up. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 15(8):771–784.

Fisher, D., Kohlhepp, P., and Bulling, F. (1998). An evolutionary algorithm for the registration of 3D surface representations. *Pattern Recognition*, 32(1):53–69.

Freeman, H. (1988). *Machine vision algorithms, architectures, and systems.* Academic Press.

Fu, K., Gonzalez, R., and Lee, C. (1987). *Robotics, control, sensing, vision, and intelligence.* Academic Press.

Fua, P. (1997). From multiple stereo views to multiple 3D surfaces. *International Journal of Computer Vision*, 24(1):19–35.

Galvez, J. and Canton, M. (1993). Normalization and shape recognition of three-dimensional objects by 3D moments. *Pattern Recognition*, 26(5):667–681.

Gonzalez, R. and Woods, R. (2001). *Digital image processing.* Second Edition. Addison-Wesley Press, U.S.

Gonzalez-Linares, J., Guil, J., and et al (2000). Hierarchical structure and nongrid motion recovery from 2D monocular views. In *Int'l Conf. Image Processing*, pages 780–783.

Gross, A. and Boult, T. (1988). Error of fit measures for recovering parametric solids. In *Int'l Conf. Computer Vision*, pages 690–694.

Gross, A. and Boult, T. (1996). Recovery of SHGCs from a single intensity view. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 18(2):161–180.

Grzeszczuk, R. and Levine, D. (1997). Brownian strings: segmenting images with stochastically deformable contours. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 19(10):1100–1114.

Guest, E., Berry, E., Baldock, R. A., Fidrich, M., and Smith, M. (2001). Robust point correspondence applied to two- and three-dimensional image registration. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 23(2):165–179.

Guillemin, V. and Pollack, A. (1974). *Differential topology.* Englewood Cliffs, Prentice-Hall, NJ.

Gunn, S. and Nixon, M. (1997). A robust snake implementation: a dual active contour. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 19(1):63–68.

Gupta, A. and Bajcsy, R. (1992). Surface and volumetric segmentation of range images using biquadrics and superquadrics. In *Int'l Conf. Pattern Recognition*, volume 1, pages 158–162.

Gupta, A. and Bajcsy, R. (1993). Volumetric segmentation of range images of 3D objects using superquadric models. *Computer Vision, Graphics and Image Processing-Image Understanding*, 58(3):302–326.

Hager, G. (1994). Task-directed computation of qualitative decisions from sensor data. *IEEE Trans. Robotics and Automations*, 10(4):415–429.

Hamann, B. (1993). Curvature approximation for triangulated surfaces. *Computing Suppl.*, 8:139–153.

Han, S., Goldgof, G., and Bowyer, K. (1993). Using hyperquadrics for shape recovery from range data. In *Int'l Conf. Computer Vision*, pages 492–496.

Hanson, A. (1988). Hyperquadrics:smoothly deformable shapes with complex polyhedral bounds. *Computer Vision, Graphics, and Image Processing*, 44:191–210.

Haralick, R. and Shapiro, L. (1993). *Computer and robot vision, volume 2*. Springer-Verlag Press, Berlin.

Hardt, V., Husson, R., and Wolf, D. (1998). An automatic calibration method for a multisensor system: Application to a mobile robot localization system. In *Int'l Conf. on Robotics and Automation*, pages 3141–3146.

Hashemi, K., Hurst, P., and Oliver, J. (1994). Sources of error in a laser rangefinder. *Review of Scientific Instruments*, 65(10):3165–3171.

Hebert, M. and Krotkov, E. (1991). 3-D measurements from imaging laser radar: How good are they? In *IEEE International Workshop on Intelligent Robots and Systems*, pages 159–164.

Heckbert, P. (1983). Texture mapping polygons in perspective. Technical Report TM 13, NYIT Computer Graphics Lab.

Hilton, A., Stoddart, A., Illingworth, J., and Windeatt, T. (1996). Reliable surface reconstruction from multiple range images. In *Proc. the 4th European Conf. Computer Vision*, pages 117–126.

Hoffman, D. and Richards, W. (1984). Parts of recognition. *Cognition*, 18:65–96.

Hoover, A., Goldgof, D., and Bowyer, K. (1998). The space envelop: a representation for 3D scenes. *Computer Vision and Image Understanding*, 69(3):310–329.

Hoover, A., Jean-Baptiste, G., Jiang, X., and etc (1996). An experimental comparison of range image segmentation algorithms. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 18(7):673–689.

Hoppe, H. and Derose, T. (1992). Surface reconstruction from unorganized points. *Computer Graphics*, 26(2):71–78.

Horn, B. (1988). *Robot vision*. MIT Press, Cambridge, MA.

Horn, B., Hilden, H., and Negahdaripour, S. (1988). Closed-form solution of absolute orientation using orthonormal matrices. *Journal of the Optical Society of America A (Optics and Image Science)*, 5(7):1127–1135.

Hu, Y. and Wee, W. (1995). Robust 3D part extraction from range images with deformable superquadric models. In *Proc. SPIE Conf. Signal Processing, Sensor Fusion and Target Recognition*, pages 524–535.

Huang, J. and Meng, C. (2001). Automatic data segmentation for geometric feature extraction from unorganized 3-D oordinate points. *IEEE Trans. Robotics and Automation*, 17(3):268–279.

Huber, D. (2001). Automatic 3D modeling using range images obtained from unknown viewpoints. In *Third Int'l Conf. on 3-D Digital Imaging and Modeling*, pages 153–160.

Imaging, Robotics and Intelligent Systems Laboratory (2002). http://imaging.utk.edu.

Integrated Vision Products (2000). *User documentation: MAPP2500 ranger PCI system, version 1.6*. Sweden.

INUS Technology, Inc (2001). *3D Reverse modeler: Rapidform 2001*. http://www.rapidform.com.

Jain, R. and Jain, A. (1990). *Analysis and interpretation of range images*. Springer-Verlag, Berlin.

Jaklič, A. (1997). *Contruction of CAD models from range images*. Ph.D. thesis, University of Ljubljana.

Jaklič, A., Leonardis, A., and Solina, F. (2000). *Segmentation and recovery of superquadrics*. Kluwer Academic Publishers.

Jarvis, R. (1983a). A laser time-of-flight range scanner for robot vision. *IEEE Trans. Pattern Analysis and Machine Intellegence*, 5(5):505–512.

Jarvis, R. (1983b). A perspective on range finding techniques for computer vision. *IEEE Trans. Pattern Analysis and Machine Intellegence*, 5(2):122–139.

Jiang, X. (2000). An adaptive contour closure algorithm and its experimental evaluation. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 22:1252–1265.

Johannesson, M. (1994). Fast, programmable, sheet-of-light range finder using mapp2200. In *Proc. SPIE Conf. Ultrahigh- and High-Speed Photography, Videography, and Photonics*, pages 25–34.

Johnson, A. and Hebert, M. (1998). Surface matching for object recognition in complex three-dimensional scenes. *Image and Vision Computing*, 16:635–651.

Johnson, A. and Hebert, M. (1999). Using spin images for efficient object recognition in cluttered 3D scenes. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 21(5):433–449.

Johnson, A. and Kang, S. (1999). Registration and integration of textured 3-D data. *Image and Vision Computing*, 17:135–147.

Kakadiaris, I., Metaxas, D., and Bajcsy, R. (1994). Active part-decomposition, shape and motion estimation of articulated objects: a physics-based approach. In *Int'l Conf. Computer Vision and Pattern Recognition*, pages 980–984.

Kanade, T. (1987). *Three-dimensional machine vision*. Kluwer Academic Publishers.

Kass, M., Witkin, A., and Terzopoulos, D. (1988). Snakes: active contour models. *International Journal of Computer Vision*, pages 321–331.

Kervrann, C. and Heitz, F. (1994). A hierarchical statistical framework for the segmentation of deformable objects in image sequences. In *Int'l Conf. Computer Vision and Pattern Recognition*, pages 724–728.

Kim, H. and Hong, K. (2000). A practical self-calibration of rotating and zooming cameras. In *Int'l Conf. Pattern Recognition*, pages 354–357.

Kim, J. and Kweon, I. (2000). A new camera calibration method for robotic applications. In *Int'l Conf. on Intelligent Robots and Systems*, pages 778–783.

Koara, K., Nishikawa, A., and Miyazaki, O. (2000). Hierarchical part decomposition method of articulated body contour. In *Int'l Conf. Intelligent Robots and systems*, volume 3, pages 2055–2060.

Koenderink, J. and Doorn, A. V. (1982). The shape of smooth objects and the way contours end. *Perception*, 11:129–137.

Koivunen, V. and Bajcsy, R. (1995). Spline representations in 3-D vision. In *Object representation in computer vision*, pages 177–190. Springer-Verlag, U.S.

Krishnamurthy, V. and Levoy, M. (1996). Fitting smooth surfaces to dense polygon meshes. *Proc. Computer Graphics (SIGGRAPH)*, pages 313–324.

Krsek, P., Lukacs, G., and Martin, R. (1998). Algorithms for computing curvatures from range data. In *The Mathematics for Surfaces VIII, Information Geometers*, pages 1–16.

Kumar, S. and Goldgof, D. (1995). Model based part segmentation of range data-hyperquadrics and dividing planes. In *Physics-Based Modeling in Computer Vision*, pages 17–22.

Kumar, S. and Goldof, D. (1996). Recovery of global nonrigid motion-a model based approach without point correspondences. In *Int'l Conf. Computer Vision and Pattern Recognition*, pages 594–599.

Kweon, I., Hoffman, R., and Krotkov, E. (1991). Experimental characterization of the perceptron laser rangefinder. Technical Report CMU-RI-TR-91-1, Carnegie Mellon University.

Leonardis, A., Gupta, A., and Bajcsy, R. (1995). Segmentation of range images as the search for geometric parametric models. *International Journal of Computer Vision*, 14:253–277.

Leonardis, A., Jaklič, A., and Solina, F. (1997). Superquadrics for segmentation and modeling range data. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 19(11):1289–1295.

Lester, E. (1998). Feature extraction, image segmentation, and surface fitting: the development of a 3D scene reconstruction system. Master's Thesis, University of Tennessee, Knoxville.

Leyton, M. (1988). A process-grammar for shape. *Artificial Intelligence*, 34:213–247.

Liang, X. and Wyvill, B. (2001). Hierarchical implicit surface refinement. In *Computer Graphics International*, pages 291–298.

Lin, C. and Perry, M. (1982). Shape description using surface triangulation. In *Computer Vision: Representation and Control*, pages 38–43.

Liu, L. and Sclaroff, S. (2001). Region segmentation via deformable model-guided split and merge. In *Int'l Conf. Computer Vision*, pages 98–104.

Loncaric, S. (1998). A survey of shape analysis techniques. *Pattern Recognition*, 31(8):983–1001.

Lorenson, W. and Cline, H. (1987). Marching cubes: a high resolution 3D surface construction algorithm. *Proc. Computer Graphics (SIGGRAPH)*, 21:163–169.

Mangan, A. and Whitaker, R. (1999). Partitioning 3D surface meshes using watershed segmentation. *IEEE Trans. Visualization and Computer Graphics*, 5(4):308–321.

Martin, R. (1998). Estimation of principal curvatures from range data. *International Journal of Shape Modeling*, 4(3):99–110.

Maver, J. and Bajcsy, R. (1993). Occlusions as a guide for planning the next view. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 15(5):417–433.

McInerney, T. and Terzopoulos, D. (1996). Deformable models in medical image analysis: a survey. *Medical image analysis*, 1(2):91–108.

Meer, P., Mintz, D., and Rosenfeld, A. (1991). Robust regression methods for computer vision: a review. *International Journal of Computer Vision*, 6(1):59–70.

Metaxas, D. (1997). *Physics-based deformable models: applications to computer vision, graphics and medical imaging*. Kluwer Academic Press, U.S.

Metaxas, D. and Dickinson, S. (1993). Integration of quantitative and qualitative techniques for deformable model fitting from orthographic, perspective, and stereo projections. In *Int'l Conf. Computer Vision*.

Metaxas, D. and Terzopoulos, D. (1993). Shape and non-grid motion estimate through physics-based synthesis. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 15(6):580–591.

Mohan, R. and Nevatia, R. (1989). Using perceptual organization to extract 3D structures. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 11(11):1121–1139.

Mohr, R. and Bajcsy, R. (1983). Packing volumes by spheres. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 5:111–116.

Moini, A. (1997). Vision chips or seeing silicon. Technical report, Centre for high performance integrated technologies and systems, the University of Adelaide.

Muraki, S. (1991a). Volumetric shape description of range data using "blobby model". *Proc. Computer Graphics (SIGGRAPH)*, 25(4):227–235.

Muraki, S. (1991b). Volumetric shape description using blobby model. *Computer Graphics*, 25(4):227–235.

National Energy Technology Laboratory, deactivition & decommissioning focus area (2002). http://www.fetc.doe.gov/dd.

Nguyen, Q. and Levine, M. (1996). Representing 3D objects in range images using geons. *IEEE Trans. Computer Vision and Image Understanding*, 63(1):158–168.

Ning, P. and Bloomenthal, J. (1993). An evaluation of implicit surface tilers. *Computer Graphics*, 13:33–41.

Nitzan, D. (1989). Three-dimensional vision structure for robot applications. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 10:291–309.

O'Donnel, T., Boult, T., and Gupta, A. (1996). Global models with parametric offsets as applied to cardiac motion recovery. In *Int'l Conf. Computer Vision and Pattern Recognition*, pages 293–299.

O'Donnel, T., Gupta, A., Fang, X., and Boult, T. (1994). The extruded generalized cylinder: a deformable model for object recovery. In *Int'l Conf. Computer Vision and Pattern Recognition*, pages 174–181.

O'Rourke, J. and Badler, N. (1979). Decomposition of three-dimensional objects into spheres. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 1:295–305.

Pal, N. and Pal, S. (1993). A review of image segmentation techniques. *Pattern Recognition*, 26(9):1277–1294.

Papaioannou, G., Karabassi, E., and Theoharis, T. (2000). Segmentation and surface characterization of arbitrary 3D meshes for object reconstruction and recognition. In *Int'l Conf. Pattern Recognition*, volume 1, pages 734–737.

Park, J., Metaxas, D., and Young, A. (1994). Deformable models with parameter functions: application to heart-wall modeling. In *Int'l Conf. Computer Vision and Pattern Recognition*, pages 437–442.

Pentland, A. (1981). Part segmentation for object recognition. *Neural Computation*, 1:82–91.

Pentland, A. (1986). Perceptual organization and the representation of natural form. *Artificial Intelligence*, 23(3):215–222.

Pentland, A. (1987). Recognition by parts. In *Int'l Conf. Computer Vision.*

Pentland, A. (1990). Automatic extraction of deformable part models. *International Journal of Computer Vision*, 4:107–126.

Pentland, A. and Sclaroff, S. (1991). Closed-form solution for physically based shape modeling and recognition. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 13(7):715–729.

Pilu, M., Fitzgibbon, A., and Fisher, R. (1996). Training PDMs on models: the case of deformable superellipses. In *The British Machine Vision Conference*, pages 373–382.

Pito, R. (1995). Characterization, calibration, and use of the Perceptron laser range finder in a controlled environment. Technical report, GRASP Lab, University of Pennsylvania.

Pito, R. (1999). A solution to the next best view problem for automatical surface acquisition. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 21(10):1016–1030.

Polleyfeys, M. and Gool, L. V. (1999). Stratified self-calibration with the modulus constraint. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 21(8):707–724.

Preparata, F. and Shamos, M. (1985). *Computational geometry-an introduction.* Springer-Verlag Press, Berlin.

Press, W., Vetterling, W., Teukolsky, S., and Flannery, B. (1992). *Numerical recipes in C: the art of scientific computing.* Cambridge Press, U.S.

Pulli, K. (1999). Multiview registration for large data sets. In *Int'l Conf. Recent Advances in 3D Digital Imaging and Modeling*, pages 160–168.

Raja, N. and Jain, A. (1992). Recognizing geons from superquadrics fitted to range data. *Image and Vision Computing*, 10(3):179–190.

Raja, N. and Jain, A. (1994). Obtaining generic parts from range images using a multi-view representation. *CVGIP: Image Understanding*, 60(1):44–64.

Ramanathan, R. and Metaxas, D. (2000). Dynamic deformable models for enhanced haptic rendering in virtual environments. In *Int'l Conf. on Virtual Reality*, pages 31–35.

Rao, K. and Nevatia, R. (1988). Computing volume descriptions from sparse 3-D data. *International Journal of Computer Vision*, 2(1):33–50.

RIEGL Laser Measurement Systems (2000). *Laser mirror scanner LMS-Z210, Technical documentation and user's instructions.* Austria.

Rivlin, E., Dickinson, S., and Rosenfeld, A. (1995). Recognition by Parts. Technical report, Center for Cognitive Science, Rutgers University.

Rom, H. and Medioni, G. (1994). Part decomposition and description of 3D shapes. In *Int'l Conf. Pattern Recognition*, pages 629–632.

Rössl, C., Kobbelt, L., and Hans-Peter, S. (2000). Extraction of feature lines on triangulated surfaces using morphological operators. In *Proceedings of the AAAI Symposium on Smart Graphics*, pages 71–75.

Roth, G. (1999). Registering two overlapping range images. In *Second Int'l Conf. on 3-D Digital Imaging and Modeling*, pages 191–200.

Rusinkiewicz, S. and Levoy, M. (2001). Efficient variants of the ICP algorithm. In *Int'l Conf. Recent Advances in 3D Digital Imaging and Modeling*, pages 145–152.

Rutishauser, M., Stricker, M., and Trobina, M. (1994). Merging range images of arbitrarily shaped objects. In *Int'l Conf. Computer Vision and Pattern Recognition*, pages 573–580.

Sacchi, R., Poliakoff, J. F., Thomas, P. D., and Hafele, K. H. (1999). Curvature estimation for segmentation of triangulated surfaces. In *Int'l Conf. 3-D Digital Imaging and Modeling*, pages 536–543.

Sander, P. and Zucker, S. (1990). Inferring surface trace and differential structure from 3-D images. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 12(9):833–854.

Sansoni, G., Carocci, M., and Rodella, R. (2000). Calibration and performace evaluation of a 3-D imaging sensor based on the projection of structured light. *IEEE Trans. Instrumentation and Measurement*, 49(3):628–636.

Sato, H. and Binford, T. (1993). Finding and recovering SHGC objects in an edge image. *CVGIP: Image Understanding*, 57(3):346–358.

Sederberg, T. and Parry, S. (1986). Free-form deformation of solid geometric models. *Proc. Computer Graphics (SIGGRAPH)*, 20:151–160.

Sharp, G., Lee, S., and Wehe, D. (2002). ICP registration using invariant features. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 24(1):90–102.

Shen, D. and Ip, H. (1997). A hopfield neural network for adaptive image segmentation: an active surface paradigm. *Pattern Recognition Letters*, 18:37–48.

Shirai, Y. (1987). *Three-dimensional computer vision.* Springer-Verlag Press, Berlin.

Solina, F. and Bajcsy, R. (1990). Recovery of parametric models from range images:the case for superquadrics with global deformations. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 12(2):131–147.

Sonka, M., Hlavac, V., and Boyle, R. (1998). *Image Processing, Analysis, and Machine Vision.* PWS Publishing, U.S.

Soucy, M. and Laurendeau, D. (1995). A general surface approach to the integration of a set of range views. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 14(4):344–358.

Stein, F. and Medioni, G. (1992). Structural indexing: efficient 3-D object recognition. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 14:125–145.

Stoddart, A. and Hilton, A. (1996). Registration of multiple point sets. In *Int'l Conf. Pattern Recognition*, pages 40–44.

Stokely, E. and Wu, S. (1992). Surface parametrization and curvature measurement of arbitrary 3-D objects: five practical methods. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 14(8):833–840.

Sullivan, S., Sandford, L., and Ponce, J. (1994). Using geometric distance fits for 3-D object modeling and recognition. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 16(12):1183–1195.

Sun, Y., Page, D., Paik, J., Koschan, A., and Abidi, M. (2002). Triangle-mesh based surface modeling using adaptive smoothing and implicit texture integration. In *Proceedings of the Int'l Symposium on 3D Data Processing, Visualization, and Transmission*, pages 588–597.

Tarabanis, K., Allen, P., and Tsai, R. (1995). A survey of sensor planning in computer vision. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 11(1):86–104.

Tasdizen, T. and Tarel, J. (1999). Algebraic curves that work better. In *Int'l Conf. Computer Vision and Pattern Recognition*, volume 2, pages 35–41.

Taubin, G. (1991). Estimation of planar curves, surfaces, and nonplanar space curves defined by implicit equations with applications to edge and range image segmentation. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 13(11):1115–1138.

Terzopoulos, D. and Metaxas, D. (1991). Dynamic 3D models with local and global deformations:deformable superquadrics. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 13(7):703–714.

Terzopoulos, D., Witkin, A., and Kass, M. (1987). Symmetry-seeking models and 3D object reconstruction. *International Journal of Computer Vision*, 1:211–221.

Terzopoulos, D., Witkin, A., and Kass, M. (1988). Constraints on deformable models: recovering 3D shape and nongrid motion. *Artificial Intelligence*, 36:91–123.

Thirion, J. (1996). New feature points based on geometric invariants for 3D image registration. *International Journal of Computer Vision*, 18(2):121–137.

Thomas, G. and Finney, R. (1979). *Calculus and analytic geometry.* Addison-Wesley, U.S.

Thompson, W., Owen, J., Germain, H., Stark, S., and Henderson, T. (1999). Feature-based reverse engineering of mechanical parts. *IEEE Trans. Robotics and Automation*, 15:57–66.

Trucco, E., Fusiello, A., and Roberto, V. (1999). Robust motion and correspondence of noisy 3-D point sets with missing data. *Pattern Recognition Letters*, 20(9):889–898.

Turk, G. and Levoy, M. (1994). Zippered polygon mesh from range images. In *Computer Graphics (SIGGRAPH 94)*, pages 311–318.

Ulupinar, F. and Nevatia, R. (1995). Shape from contour: straight homogeneous generalized cylinders and constant cross section generalized cyliners. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 17(2):120–134.

Vemuri, B. and Guo, Y. (2000). Snake pedals: compact and versatile geometric models with physics-based control. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 22(5):445–459.

Vidmar, A. and Solina, F. (1992). Reconstruction of superquadrics from 2D contours. *Mathematics research*, 69:227–240.

VRML Model Gallery, OCNUS Company (2002). http://www.ocnus.com/models/people/.

Wang, Y., Peterson, B., and Staib, L. (2000). Shape-based 3D surface correspondence using geodesic and local geometry. In *Int'l Conf. Computer Vision and Pattern Recognition*, pages 663–668.

Wernecke, J. (1994). *The inventor mentor: programming object-oriented 3D graphics with open inventor*. Addison-Wesley, U.S.

Whaite, P. and Ferrie, F. (1991). From uncertainty to visual exploration. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 13(10):1038–1049.

Whaite, P. and Ferrie, F. (1997a). Autonomous exploration: driven by uncertainty. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 19(3):193–205.

Whaite, P. and Ferrie, F. (1997b). On the sequential determination of model misfit. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 19(8):899–905.

Whitaker, R. (1994). Volumetric deformable models: active blobs. In *Proc. SPIE Conf. Visualization in Biomedical Computing*, pages 122–134.

Whitaker, R. (1998). A level-set approach to 3D reconstruction from range data. *International Journal of Computer Vision*, 29(3):203–231.

Whitaker, R. and Gregor, J. (2001). Indoor scene reconstruction from sets of noisy range images. *Graphical Models*, 63:304–332.

Wilczkoniak, M., Boyer, E., and Sturm, P. (2001). Camera calibration and 3D reconstruction from single images using parallelepipeds. In *Int'l Conf. Computer Vision*, pages 142–148.

Witkin, A., Fleischer, K., and Barr, A. (1987). Energy constraints on parameterized models. In *Proc. SIGGRAPH.*

Wong, Y., Yuen, P., and Tong, C. (1998). Segmented snake for contour detection. *Pattern Recognition*, 31(11):169–1679.

Wu, K. and Levine, M. (1993). 3D object representation using parametric Geons. Technical Report TR-CIM-93-13, McGill University.

Wu, K. and Levine, M. (1997). 3D part segmentation using simulated electrical charge distributions. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 19(11):1223–1235.

Wu, K. and Levine, M. D. (1994). Recovering parametric geons from multiview range data. In *Int'l Conf. Computer Vision and Pattern Recognition*, pages 159–166.

Yahia, H., Huot, E., Herlin, I., and Cohen, I. (2000). Geodesic distance evoluation of surfaces: a new method for matching surfaces. In *Int'l Conf. Computer Vision and Pattern Recognition*, pages 644–651.

Yamany, S., Ahmed, M., Hemayed, E., and Farag, A. (1998). Novel surface registration using the grid closest point (GCP) transform. In *Int'l Conf. Image Processing*, pages 809–813.

Yamany, S. and Farag, A. (1999). Free-form surface registration using surface signatures. In *Int'l Conf. Computer Vision*, pages 1098–1104.

Yim, C. and Bovik, A. C. (1998). Multiresolution 3D range segmentation using focus cues. *IEEE Trans. Image Processing*, 7(9):1283–1298.

Yuan, X. (1995). A mechanism of automatic 3D object modeling. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 17(3):157–172.

Zerroug, M. and Nevatia, R. (1996). Three-dimensional descriptions based on the analysis of the invariant and quasi-invariant properties of some curved-axis generalized cylinders. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 18(3):237–253.

Zerrough, M. and Nevatia, R. (1999). Part-based 3D descriptions of complex objects from a single image. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 21(9):835–848.

Zerrough, M. and Neviata, R. (1994). Segmentation and recovery of SHGCs from a real intensity image. In *Proc. 3rd European Conf. Computer Vision*, volume 1, pages 319–330.

Zha, H., Hoshide, T., and Hasegawa, T. (1998). A recursive fitting-and-splitting algorithm for 3D object modeling using superquadrics. In *Int'l Conf. Pattern Recognition*, volume 1, pages 658–662.

Zhang, D. and Hebert, M. (1999). Harmonic maps and their applications in surface matching. In *Int'l Conf. Computer Vision and Pattern Recognition*, pages 524–530.

Zhang, Y., Koschan, A., and Abidi, M. A. (2003). Superquadrics based 3D object representation of automotive parts utilizing part decomposition. In *the 6th Int'l Conf. Quality Control by Artificial Vision (QCAV)*.

Zhang, Y., Paik, J., Koschan, A., and Abidi, M. A. (2002a). 3-D object representation from multi-view range data applying deformable superquadrics. In *Int'l Conf. Pattern Recognition*, volume 3, pages 611–614.

Zhang, Y., Paik, J., Koschan, A., and Abidi, M. A. (2002b). A simple and efficient algorithm for part decomposition of 3-D triangulated models based on curvature analysis. In *Int'l Conf. Image Processing*, volume 3, pages 273–276.

Zhang, Y., Price, J. R., and Abidi, M. A. (2001). Superquadrics-based object representation of complex scenes from range images. In *Proc. SPIE Conf. 3D Image Capture and Applications IV*, pages 56–67.

Zhang, Y., Sun, Y., Price, J. R., and Abidi, M. A. (2000). Impact of intensity edge map on segmentation of noisy range images. In *Proc. SPIE Conf. 3D Image Capture and Applications III*, volume 3958, pages 260–269.

Zhang, Z. (1994). Iterative point matching for registration of free-form curves and surfaces. *International Journal of Computer Vision*, 13:119–152.

Zhou, L. and Kambhamettu, C. (1999). Extending superquadrics with exponent functions: modeling and reconstruction. In *Int'l Conf. Computer Vision and Pattern Recognition*, volume 2, pages 73–78.

Zhou, L. and Kambhamettu, C. (2000). Hierarchical structure and nongrid motion recovery from 2D monocular views. In *Int'l Conf. Computer Vision and Pattern Recognition*, pages 752–759.

Zhuang, Y. and Canny, J. (2000). Haptic interaction with global deformations. In *Int'l Conf. on Robotics and Automation*, pages 2428–2433.

Appendix

Appendix A

Characterization of the IVP Ranger Scanner System

In this chapter, we describe the experimental characterization and analyze the accuracy of the range camera used in our experiments. In Section A.1, related work on range sensor characterization is discussed. The range sensor used in our work is introduced in Section A.2. In Section A.3, the MAPP 2500 range camera is characterized in terms of temperature, light, surface reflectance and color of objects, and the angle of incidence in the experiments. The accuracy of the captured data is also investigated by calibrating the range images into 3D data in Section A.4. Finally, Section A.5 summarizes the characteristics and accuracy of the smart range sensor based on experiments.

A.1 Related Work

Range image sensing techniques have been developed for over thirty years and are becoming more and more popular due to the decreasing cost and 3D sensing advantages. A range image is an image in which each pixel represents a measurement of the distance from the camera to the object. Typical applications of range imaging include industrial inspection and robot navigation (Jain and Jain, 1990; Adams and Robert, 1996). Range sensors are unique in that the image data points explicitly represent scene surface geometry in a sampled form (Faugeras, 1993; Kanade, 1987). Most range sensors can be classified into two major types, active and passive cameras. A passive range camera utilizes no special (dedicated) light source, and may use, for instance, information from camera and/or object motion to obtain range data. An active camera utilizes a well defined light source to gather range information. Active range sensors can generally be classified into six categories (Besl, 1989): (1) radar, (2) triangulation, (3) moire, (4) holographic interferometry, (5) lens focusing, and (6) diffraction. The first two types of active range cameras are predominately used. According to the signal type emitted by the laser, imaging radars can be classified into time-of-flight sensors which use pulse detection techniques, amplitude modulation sensors in which the laser beam is amplitude-modulated, or frequency modulation sensors which use heterodyne detection techniques. Time-of-flight methods are usually good for large distances, i.e. meters to kilometers. Triangulation based range cameras use a dedicated

light source placed in a well-known position emitting light in a well-known direction. The position in space where the light was reflected from is found by measuring where the light impacts the sensor. Triangulation based cameras work better when measuring shorter distances ($< 5 - 10\ m$). More comprehensive knowledge about laser rangefinders can be found in (Fu et al., 1987; Freeman, 1988; Jarvis, 1983b; Jarvis, 1983a; Nitzan, 1989).

Since each type of sensor is more suitable for a specific task, there are several issues to be investigated in order to optimize the performance of the sensor. The major consideration is under what circumstances can the sensor obtain the most accurate and reliable data (Hebert and Krotkov, 1991; D. Nitzan and Duda, 1977; Hashemi et al., 1994). Characterization is a process to quantitatively evaluate all the factors that affect the performance or the quality of the measured sensor data (Collier, 1998). The comprehensive knowledge obtained during the characterization process including factors that influence the measured data, the optimal sensor system configuration, the accuracy of the sensor, and other issues can be used to analyze the sensor performance, and can furthermore, maximize this performance. Therefore, we believe that the characterization of a sensor directly affects its performance optimization and is an important and indispensable step required before the sensor can be appropriately used. Despite the broad expanse of range imaging systems available today, there is only a modest amount of research being conducted on sensor characterizations compared with research on sensor calibration (Sansoni et al., 2000; Wilczkoniak et al., 2001; Kim and Kweon, 2000; Kim and Hong, 2000; Castro et al., 1998; Hardt et al., 1998; Beraldin et al., 1993). Manufacturer specifications are often accepted as a working description of a sensor system. However, they are not always complete, accurate or appropriate for specific tasks.

Kweon, at al. (Kweon et al., 1991) published one of the first characterizations of the Perceptron AM laser range scanner. They ascertained many of the sensor characteristics by observing images both spatially and temporally. A statistical analysis determines the precision of pixel positions and range measurements. The analysis associated with range measurements includes the effects of ambient light, surface material, incidence angle, and temperature. Richard Pito (Pito, 1995) applied a dynamic calibration technique to the Perceptron AM laser range finder. In this characterization procedure, micro-scale spatial range noise caused by local environments, macro-scale spatial range noise attributed by complex interactions between subsystems, mixed pixel spatial range noise, and temporal range noise were analyzed. Accuracy and repeatability of range measurements were obtained. Moreover, the effects of the scene, the surface reflectance and the distance were investigated. Conclusions on using the sensor appropriately, and the necessary warm-up time were derived. Richard Collier (Collier, 1998) characterized a point laser range finder. In his thesis, the influences of temperature, mixed pixels, reflectance, and angle of incidence were analyzed. An error correction model and a confidence metric were also developed.

Although characterizations of several kinds of range sensors have been conducted, no research on characterizations of smart range cameras has been found. Characterization of each range scanner is unique and per-system based due to the scanner's specific characteristics and the principles under which the sensor works. This work therefore presents the first attempt to characterize the smart range sensor MAPP2500, which is a sheet-of-light triangulation-based sensor. This sensor was selected due to its high speed (4000 profiles/s) and accuracy (μm)

(Johannesson, 1994). In the experiments conducted, the accuracy of the range data is analyzed utilizing calibrated 3D data. The errors present in the measured range data are investigated under circumstances with varying temperatures, lights, surface reflectance and color of objects, and incidence angles in the experiments since these factors usually affect the performance of a laser scanner (Pito, 1995). By characterizing a specific sensor, this work also makes contributions to showing how the performance of a range sensor can be evaluated and optimized by experimental characterizations.

A.2 The IVP Range Sensor

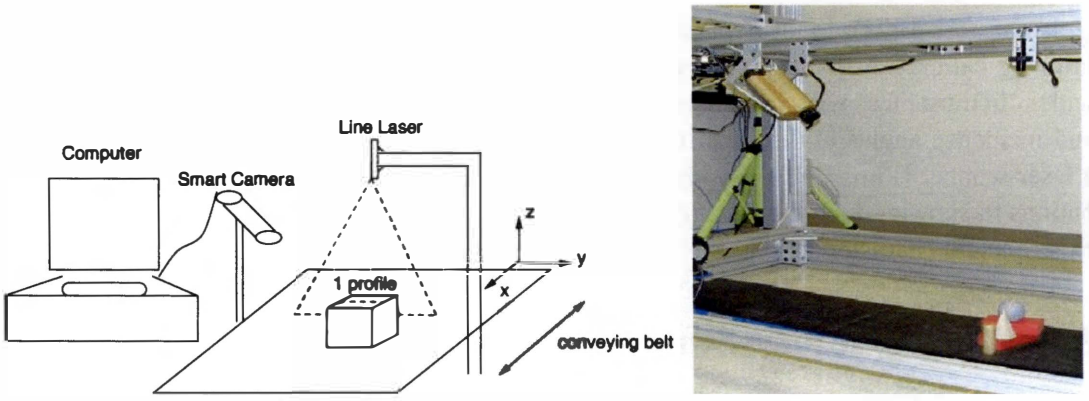
In this section the sensor system configuration and scanning principles of the smart range sensor MAPP2500 are briefly described. A smart sensor is a sensor which combines image acquisition and signal processing/data reduction on the same chip (Astrŏm, 1993), (Moini, 1997). The main signal processing task in sheet-of-light imaging is a data extraction process where 1D range information is extracted from 2D sensor data. MAPP2500 stands for Matrix Array Picture Processor with 2-Dimensional sensor and 512 processing elements (Johannesson, 1994), (Astrand et al., 1995). MAPP-serial smart range sensors have been available commercially since 1991. With a smart sensor the scanning speed is improved significantly by the special on-chip processing capabilities.

A.2.1 Introduction of the System Setup

In this research, the RANGER system package consists of a MAPP2500 smart camera, a 3D profiling software module, an application programmer's interface, and an advanced graphical user interface for Windows NT. A smart camera can mean many things to different manufacturers, and the best definition often lies within the context of a specific application. Generally, a smart sensor includes a sensor, a digital signal processor for analog-to-digital conversion, soft memory for temporary image storage and manipulation, and hard memory for operational software and algorithm storage.

In our case, the MAPP2500 smart camera consists of a smart vision sensor, an embedded processor, and a high-speed serial interface that connects the system with a host PC through a thin cable. The smart vision sensor is a combination of an image sensor array, an A/D-converter and general-purpose image processors on the same CMOS chip (Integrated Vision Products, 2000). Each individual image sensor is triangulation-based, and utilizes a sheet-of-light (light stripe). The simplified diagram of the system setup is shown in Fig. A.1(a). Resolution of a captured range image is up to 512×65536 pixels. The maximum acquisition speed of the camera is 4000 profiles per second and registered range and intensity images are captured simultaneously.

Due to the sheet-of-light utilized, the range camera only finds one range profile of the scene at each exposure, and a scanning mechanism is therefore needed to obtain a 2D range image of the scene. There are three means of scanning including: moving the objects, moving the camera or using a mirror to reflect the laser beam. In our experiments, the first method is used since the tested objects are relatively small (less than $50 \times 50 \times 50$ cm) and the vibration of the



(a) (b)

Figure A.1: Range imaging system configuration: PC, line laser source, range camera, and conveyor belt. (a) Simplified diagram of the setup and (b) photograph of the system configuration used in the experiments.

sensor system should be minimized. A horizontal conveyor belt is used to move the object at a user-selected speed. A photograph of the system configuration is shown in Fig. A.1(b).

A.2.2 Algorithms for Obtaining Range Data

The RANGER software package contains 4 basic algorithms, a vertical maximum, a vertical thresholding, a horizontal maximum, and a horizontal thresholding algorithm to capture range images. With respect to the angle between the camera orientation and the baseline, the algorithms are classified into horizontal and vertical. Corresponding plots of profiles scanned in horizontal and vertical orientations are shown in Fig. A.2. According to how the laser impact position is detected, the algorithms can also be classified into maximum and thresholding. The maximum-finding algorithm locates the impact position of the sheet-of-light by finding the position of the maximum intensity. The thresholding algorithm computes the centroid position of the thresholded data. A graphical representation of methods used for determining the impact position are shown in Fig. A.3.

For the maximum method, the real impact position is estimated as

$$s = \frac{(a+b)}{2}, \tag{A.1}$$

where a and b are offset positions of the two maximum intensities shown in Fig. A.3.

For the thresholding method, the peak mid-position is approximated by

$$s = \frac{(n+m)}{2}, \tag{A.2}$$

where m and n are also shown in Fig. A.3.

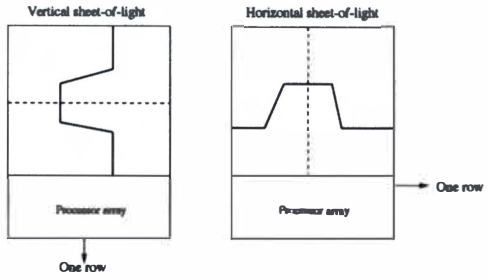


Figure A.2: Profiles scanned in vertical and horizontal sensor orientations.

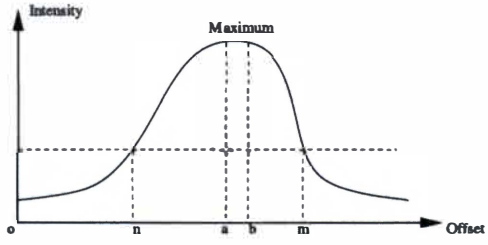


Figure A.3: Illustration of maximum and threshold algorithms. The algorithms find the laser reflection (laser impact position), detect maximum peak intensity and threshold the measurements.

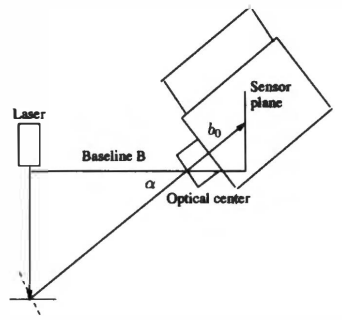


Figure A.4: Triangulation-based setup to calculate range values.

Equations. (A.1) and (A.2) result in a laser impact position halfway between two pixel centers, which also indicates that both thresholding and maximum algorithms are able to achieve sub-pixel accuracy but the thresholding algorithms are less sensitive to noise. However, since the thresholding algorithm is designed to obtain only range images, the maximum algorithm must be chosen if both range and intensity images are needed. After the camera orientation and the algorithm are selected, range values are calculated based on the triangulation rule. A triangulation-based procedure to derive range values is illustrated in Fig. A.4.

The range value r is calculated by

$$r = \frac{B(b_0 \tan\alpha - s)\cos\alpha}{\frac{b_0}{\cos\alpha} - (b_0 \tan\alpha - s)\sin\alpha} = B\frac{b_0 \tan\alpha - s}{b_0 + s\tan\alpha}, \tag{A.3}$$

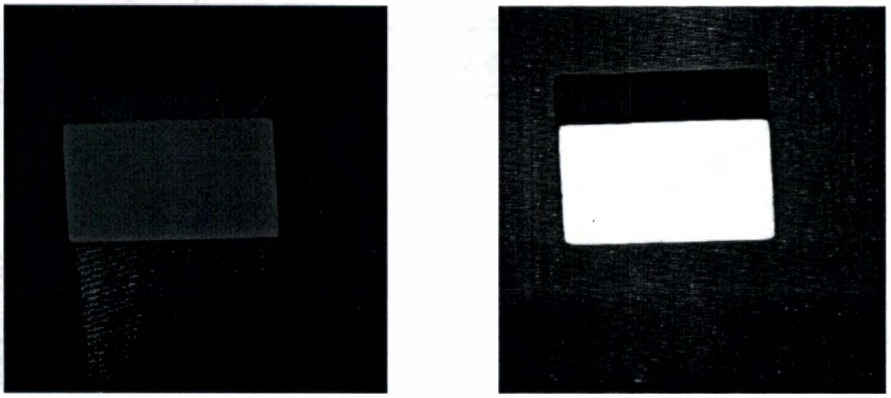
where s represents the light impact position calculated by Equations. (A.1) and (A.2). The other parameters are as illustrated in Fig. A.4.

A.3 Experimental Characterization

Although characterization of each range scanner is unique due to its specific characteristics and the principles under which the sensor works, part of the characterization scheme can be shared between various laser scanners. Similar to the characterization framework presented in (Pito, 1995), in this section the smart range camera is characterized with respect to the following five issues; (1) how the captured range data fluctuates with temperature, (2) how environmental light affects the data, (3) how the surface characteristics of the object itself including its surface reflectance and color, influence the data, and (4) whether the angle of incidence affects the range measurements.

A.3.1 Initialization

The orientation of the camera and the laser is determined according to the horizontal application selected. The laser is placed vertical to the ground, the angle between the optical axis of the



(a) (b)

Figure A.5: Range and intensity images of upper surfaces of objects. Images are taken under medium indoor light condition. (a) Range image and (b) intensity image.

camera and the baseline is 45^0. The conveyor belt is placed on the ground horizontally. The RANGER system is set up as shown in Fig. A.1(b). The horizontal threshold algorithm is used during the characterization process due to its robustness to noise and fast scans. The major parameters of the algorithm are chosen heuristically depending on the size of the object scanned, the accuracy required by users, and the speed of the conveyor belt. The objects used include a red wood block, a black wood block, and a metal block of the same size at $10 \times 6 \times 3$ cm. All the objects used have matte surfaces, and there is no large intensity variations on those surfaces. The range and intensity images of the red wood block used are shown in Fig. A.5.

Only the upper flat surface of the blocks is scanned during the characterization process. Range images captured under various conditions are analyzed by calculating and comparing the mean values, the standard deviations, and the maximum values of the pixel intensities in the images. There are no physical units for these statistical index due to the rescaled intensities values used.

The mean value μ of the samples is calculated by

$$\mu = \frac{\sum_{i=1}^{N} x_i}{N}, \tag{A.4}$$

where N is the number of non-zero pixels in the captured range image, x_i denotes non-zero range values of pixel i. The standard deviation σ of the samples is calculated by

$$\sigma = \sqrt{\frac{\sum_{i=1}^{N}(x_i - \mu)^2}{N}}. \tag{A.5}$$

The maximum value of each captured range image is

$$Max = Maximum(x_i), i = 1, 2, ...N. \tag{A.6}$$

Because only the upper flat surfaces are scanned in our experiments, for each scanning circumstance the mean value should be very close to the maximum value, and σ should be 0 in the ideal case. This is one of the criteria used to evaluate the accuracy of the measured range images.

A.3.2 Effects of Temperature

The fluctuation of range measurements caused by temperature is common to all laser range sensors and a warm-up time is therefore necessary for all laser scanners. However, since the measurement fluctuation and warm-up time vary with sensors a characterization procedure is needed to evaluate the accurate warm-up time for our sensor to obtain stable and reliable measurements. Temperature generally has two sources. The first is environmental temperature. Manufacturers usually provide a span of such environmental temperatures that their laser and sensor should be operated in to avoid significant inaccuracies in range measurements. Because this sensor is designed to be used mainly in an indoor environment, it is assumed that fluctuations due to environmental temperature are limited and are therefore ignored. The second source of temperature fluctuation is device electronics. When the system initializes from a "cold" state, time is required for the temperature in the device electronics to stabilize. To estimate the necessary warm-up time, 10 images are captured every 30 minutes during a 2-hour period, starting immediately after the laser and the system are powered. The goal is to estimate the warm-up time required by the system, and to investigate how the range data is affected by the temperature of the device electronics. To achieve this goal, the mean values, the standard deviations, and the maximum values are calculated for each image and their plots are shown in Fig. A.6.

Based on the plots in Fig. A.6 (a) and (b), the two image sets taken during the second hour have closer mean values (the curves are less steep) between each other and smaller standard deviations (13.5 vs. 18) than images taken during the first hour. Moreover, the images captured during the second hour have closer mean and maximum values (106, 110) than those taken during the first hour (107, 115). Therefore, images taken in the second hour are more stable and reliable. The conclusion can then be ascertained that there should be at least 60 minutes for the system to warm up and obtain reliable range measurements. Based on this finding, all the other range images used in our experiments were taken after warming up the sensor for 60 mins.

A.3.3 Effects of Environmental Light

The fact that range scanners are sensitive to environmental light is well known. To answer questions such as how sensitive a specific sensor is to various light conditions and how the noise caused by environmental light can be reduced, an experimental characterization is required. This characterization investigates how the range images are affected by environmental light. Since this camera is designed mainly for indoor applications, 10 images are taken under each of the three indoor light conditions; 3 lumen (very dark), 500 lumen (medium), and 3000 lumen (very bright). The mean value, the standard deviation, and the maximum value of each image taken under different light conditions are calculated and their plots are shown in Fig. A.7. The graphs in Fig. A.7 show that the images taken under very dark and medium light conditions have very close mean values (around 108), standard deviations (around 14), and maximum values

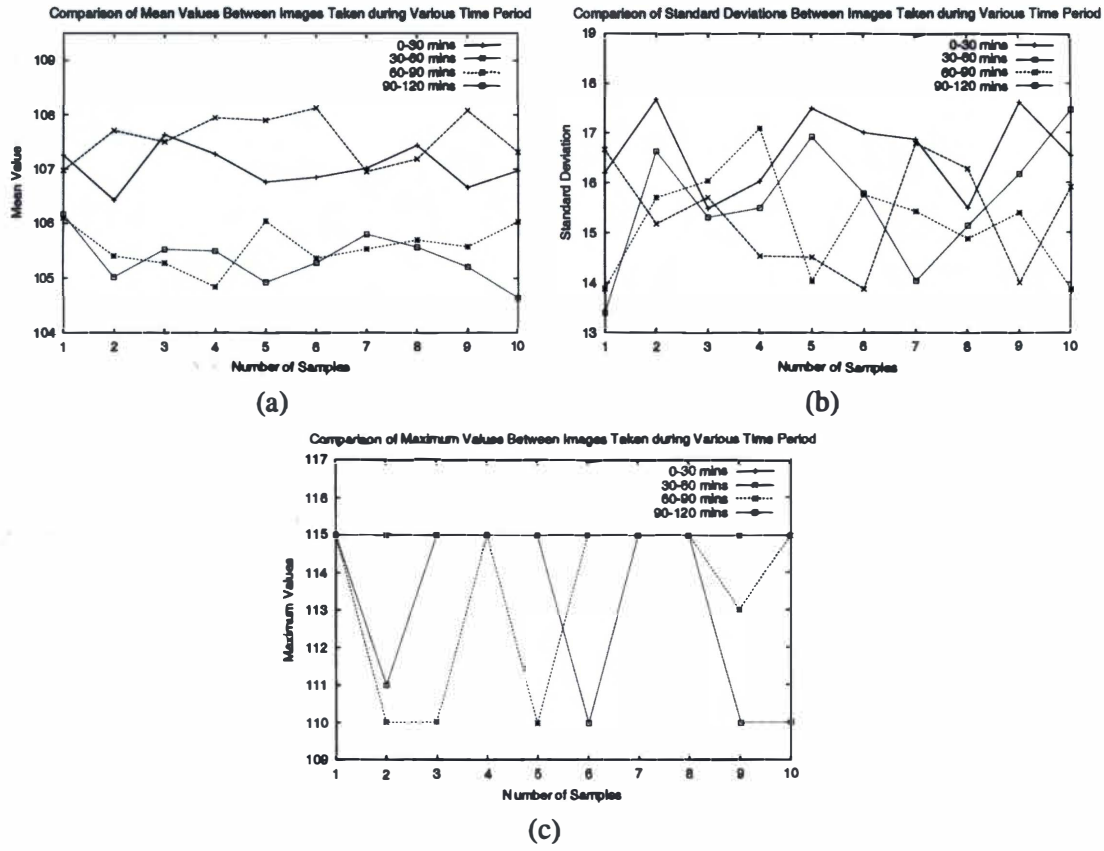


Figure A.6: Influence of temperature: mean values, standard deviations, and maximum values of 4 sets of range images taken every 30 minutes during a 2-hour time period. (a) Mean values, (b) standard deviations, and (c) maximum values.

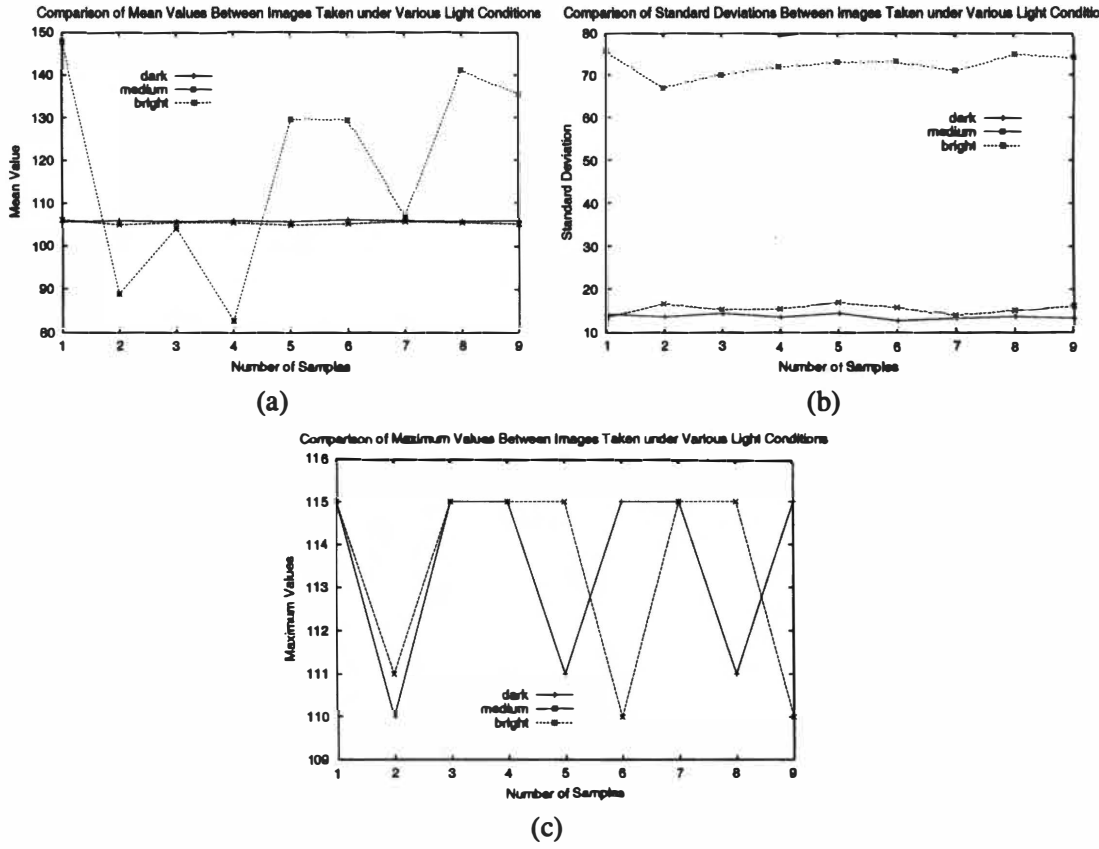
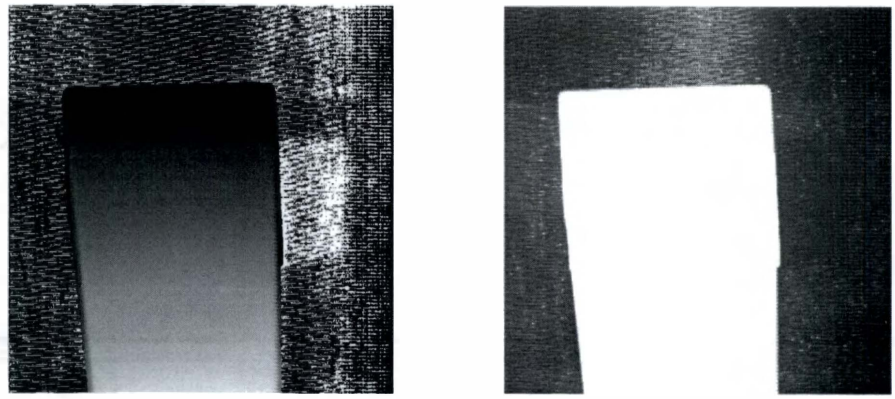


Figure A.7: Influence of light: comparison of mean values, standard deviations, and maximum values of 3 sets of range images taken under very dark, normal, and very bright light conditions. (a) Mean values, (b) standard deviations, and (c) maximum values.



(a) (b)

Figure A.8: Range and intensity images under very bright light condition (3000 lumen). (a) Range image and (b) intensity image.

(110-115). In addition, the mean values are closer to the maximum values (108 vs. 110) for images taken under very dark and medium conditions. However, the features of images taken in very bright conditions are very different. For these images, the mean values range from 80 to 150, the standard deviations are as high as 80, and the maximum values (around 115) are far from the corresponding mean values. An example of range and intensity images taken in very bright conditions is shown in Fig. A.8. Compared with images in Fig. A.5 and combining the statistical results obtained, the conclusion is drawn that this range sensor is sensitive to light, and tends to have incorrect range and intensity measurements when the environmental light is very strong ($>$ 500 lumen). Moreover, the best light condition for a specific sensor varies with the power of the laser used. In our case, the power output of the laser is 5 mW, and the wavelength is 685 nm. Therefore, it is recommended that the environmental light should be as low as possible to acquire accurate data.

A.3.4 Effects of Surface Reflectance and Color of Objects

Since active range sensors acquire range data by emitting a laser beam to surfaces of an object the range measurements are inevitably affected by the reflectance and other surface characteristics of the object. However, the effects may vary with different active range sensors. To investigate how the surface characteristics of the objects affect the range images captured by our sensor, the influences of surface reflectance and color are characterized respectively by experimentation.

First, to investigate the influence of surface reflectance, 5 images are captured for each a metal and a wooden block of the same size. The mean value, the standard deviation, and the maximum value of each image taken for the metal and wooden blocks are calculated and shown in Fig. A.9. The graphs in Fig. A.9 (a) and (b) illustrate that images of metal and wooden blocks have slightly different mean values (wood: around 125, metal: around 119). The standard deviations of wood (around 30) are smaller than those of metal (around 33), which means that

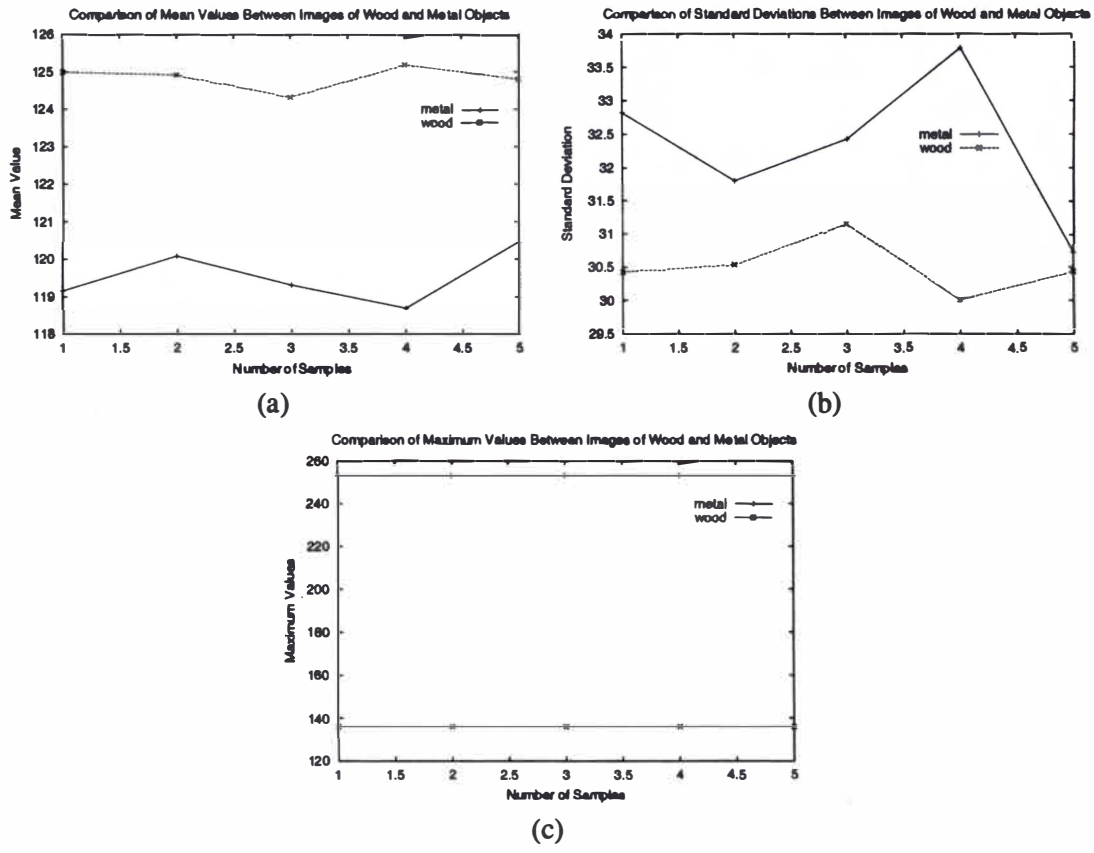


Figure A.9: Influence of surface reflectance of objects: comparison of mean values, standard deviations, and maximum values between 3 sets of range images taken for metal and wooden blocks. (a) Mean values, (b) standard deviations, and (c) maximum values.

images of wood are more stable. The maximum values captured from these two blocks are significantly different (metal 245, wood 135). The significant difference could come from the noise caused by the non-lambertian surface reflectance of the metal. Also, for the wooden block, the mean values are closer to the maximum values while there are large differences for the metal block. The conclusion can be made that this range sensor is not very sensitive to the surface reflectance of objects, but heavier noise is apparent for non-lambertian surfaces with high reflectance such as metal surfaces.

Next, to investigate the influence of the surface color on the captured range image, a red block, a black block of the same size, and two blocks joined together are used to take 5 images of each. The mean value, the standard deviation, and the maximum value of each image are calculated and shown in Fig. A.10. The graphs in Fig. A.10 demonstrate that images of red blocks have significantly different features including the mean values (108 vs. 52), the standard deviation values (13 vs. 45), and the maximum values (115 vs. 125) from those of black blocks. The differences are obvious even though the blocks used have the same dimensions. This is because the laser beam reflectance features of black surfaces are different from those of other colors. Furthermore, for the red block, the mean values are very close to the maximum values (108 vs. 115) while there are large differences for the black block (52 vs. 125). Therefore, the conclusion can be drawn that range measurements of completely black objects should be corrected before use. The correction could be adjusting the environmental light or covering the object surface with other lighter colors.

A.3.5 Effects of Incidence Angle

The incidence angle is another substantial contributor to the error present in laser range data because rangefinders rely on the detection of reflected signals. If the signals are intersecting with a target at an angle, a portion of the signal will not be reflected back in the direction of the detector. However, in many cases the incidence angle cannot be 0, and whether the measured data are acceptable within a range of incidence angles needs to be investigated. The angle of incidence is defined as the angle between the laser direction and the normal of the measured surface. An example of incidence angle is shown in Fig. A.11.

Since the laser is fixed as being vertical to the ground for all scanning applications, the incidence angle varies only with the normal of the measured surface. The angle ranges from 0^0 for a flat horizontal surface, to 90^0 for a vertical surface. The measurement error increases with increasing incidence angle. In the experiments conducted, range measurements are obtained for a flat surface with incidence angles at 0^0 and 30^0, respectively. The range images, the corresponding calibrated 3D data, and the zoomed wire frames of the 3D data are shown in Fig. A.12. Despite the accuracy analysis in Section A.4, the wire frames of the incidence angle at 0^0, shown in Fig. A.12 (c) are smoother and flatter, while those of the incidence angle at 30^0, shown in Fig. A.12 (f) have more ridges.

This demonstrates that the incidence angle does have effect on the range measurements and that the range measurements of smaller incidence angles are more accurate than those of larger incidence angles. Whether the measured data with the incidence angle at 30^0 are acceptable depends on later accuracy analysis of 3D calibrated data.

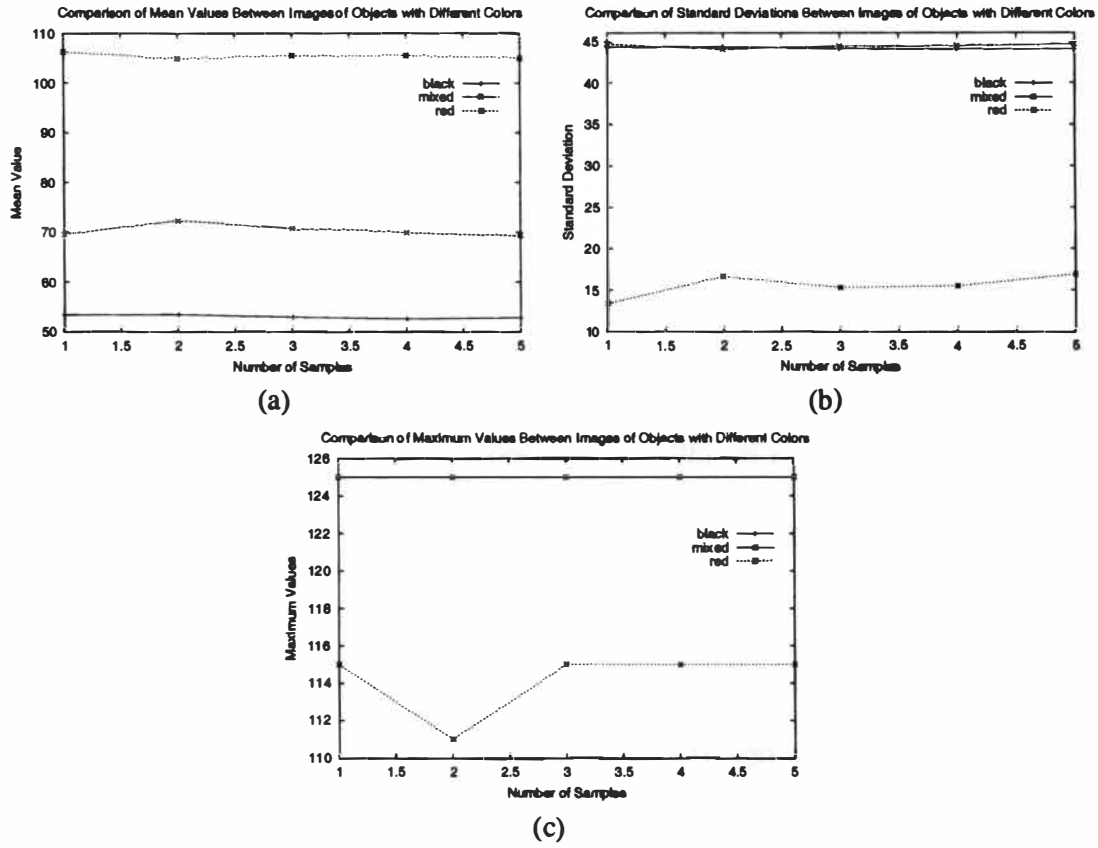


Figure A.10: Influence of color: comparison of mean values, standard deviations, and maximum values between 3 sets of range images of red and black wooden blocks. (a) Mean values, (b) standard deviations, and (c) maximum values.

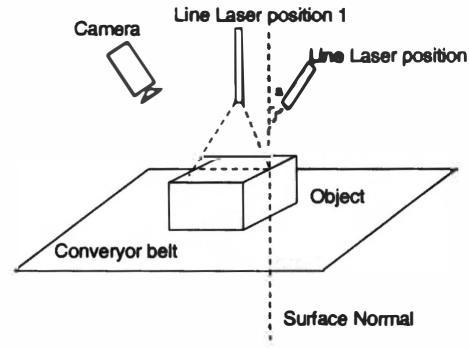


Figure A.11: Incidence angle example. The incidence angle is 0 for laser position 1 and 'a' for laser position 2.

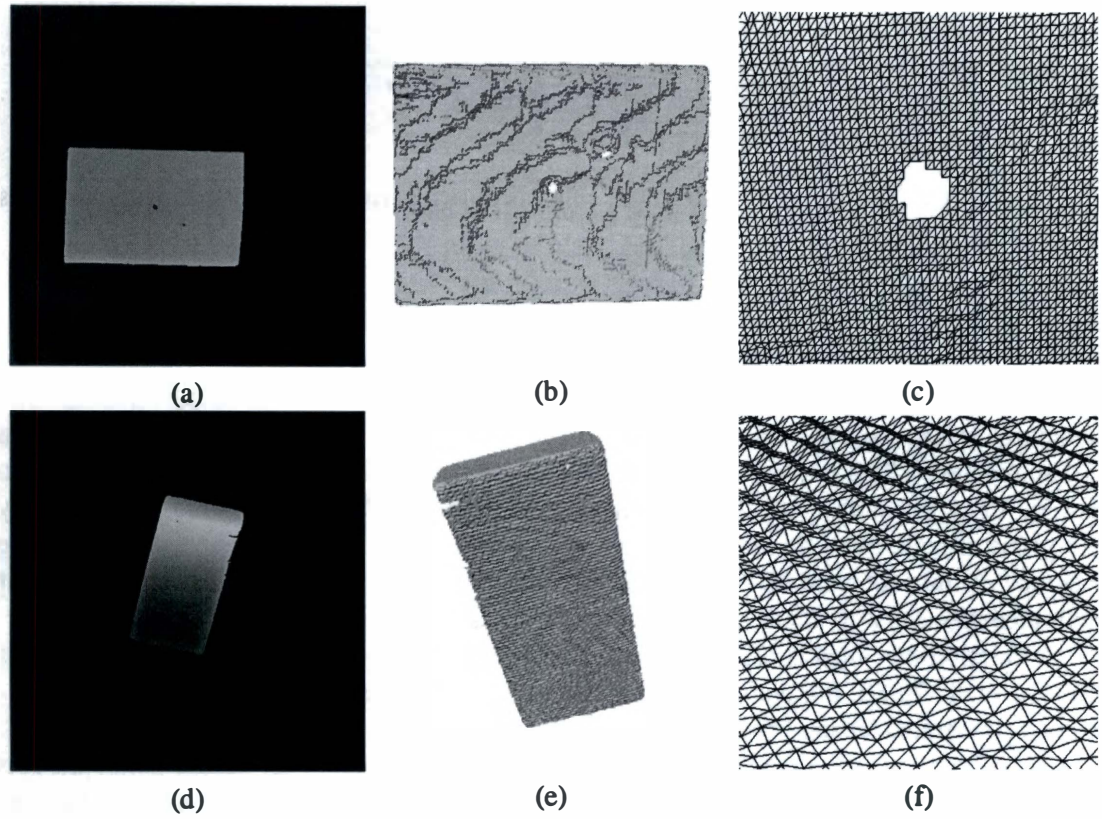


Figure A.12: Captured range and corresponding calibrated 3D data. (a) Range image of block a, the incidence angle of the upper flat surface is 0^0, (b) calibrated 3D data of block a, (c) zoomed wire frames of calibrated 3D data: very smooth, (d) range image of block b, the incidence angle of the upper flat surface is 30^0, (e) calibrated 3D data of block b, and (f) zoomed wire frames of calibrated 3D data: many ridges.

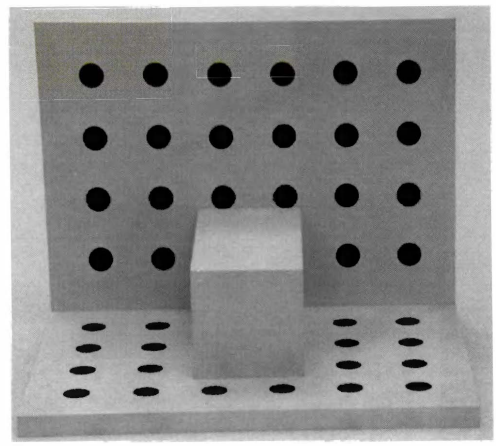


Figure A.13: Calibration target. The target consists of two perpendicular planes with dark dots distributed uniformly.

A.4 Accuracy Analysis

Generally, the maximum accuracy claimed in product manuals is very difficult to achieve in real applications. To analyze the accuracy, the 2D range images need to be calibrated to obtain real 3D (x, y, z) coordinates for each image pixel since 3D data provides the real geometry and physical features of the object. In this section, calibration is first conducted and then the accuracy of the calibrated 3D data is analyzed. Simultaneously, the accuracy investigation of calibrated data can also check the correctness of the calibration procedure.

Calibration of cameras varies with different principles under which the camera operates (Sansoni et al., 2000). For our camera, the calibration target was built as suggested in the user's manual (Integrated Vision Products, 2000), and is shown in Fig. A.13.

The calibration target consists of two $200 \times 150\ mm$ perpendicular planes with dark dots distributed uniformly on the target. The center of gravity of these dots will be used to calculate the calibration parameters including three matrices, $W[4]$, $P[3][3]$, and $T[3]$. W is the scaling of the sensor coordinate system. P represents the transform corresponding to the lens combined projection, sensor scaling and rotation and T is the translation matrix. Further details of the calibration parameters and algorithm can be found in (Integrated Vision Products, 2000). The calibration model used in the RANGER system is based on well-known geometric camera modeling techniques (Faugeras, 1993). Detailed calibration steps are described in (Integrated Vision Products, 2000). Several important factors that are not addressed by the manuals in the calibration procedure are described in the following.

- The focus and aperture of the camera should be adjusted depending on the distance between the baseline and the object to make sure the intensity image is visibly adequate.

- The illumination of the calibration target should be adjusted appropriately by either adjusting the camera lens or adding additional light. The surface of the target should not have high reflectance, i.e., it must be dull and matte.
- The world coordinate system used in the calibration process should be selected as the X-axis consistent with the direction of the profile, the Y-axis consistent with the scanning direction, and the Z-axis determined by the right-hand rule.
- The Y-scaling parameter depends on the moving speed of the conveyor belt, and can be estimated by evaluating the ratio between the range values and the 3D coordinates of the measured object. The Y-scaling parameter is used together with the other calibration parameters to convert 2D range images into 3D calibrated range data.

The calibration parameters during our accuracy evaluation process are obtained as

$$W = \begin{bmatrix} 0.000008561298 \\ 0.000485681543 \\ -0.792188565246 \\ -514.894369030070 \end{bmatrix},$$

$$T = \begin{bmatrix} 88.329116344283 \\ 498.478733669282 \\ 593.623597718181 \end{bmatrix},$$

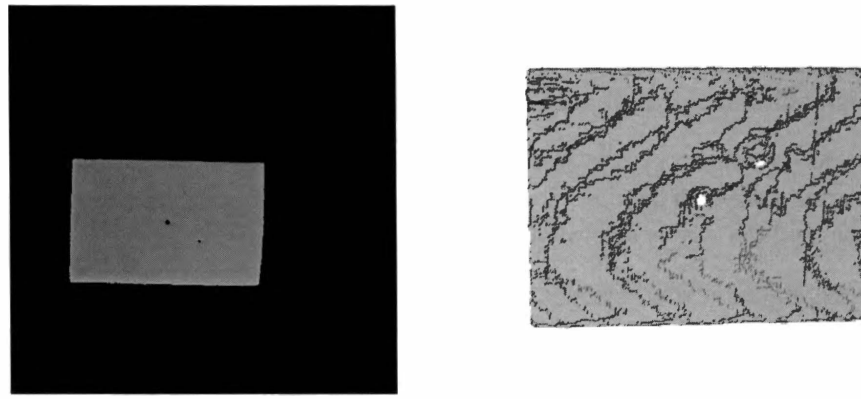
$$P = \begin{bmatrix} -0.000662504 & 0.000011954 & 0.181907633 \\ 0.000007590 & -0.000485012 & 0.789550350 \\ 0.000006133 & 0.000447274 & 0.628674025 \end{bmatrix},$$

and

$$Yscaling = 0.53, beltspeed = 70mm/s.$$

Fig. A.14 shows a captured 2D range image and corresponding calibrated 3D data of the upper surface of a block. The calibrated data in Fig. A.14(b) shows that the camera is calibrated correctly. To investigate the accuracy of the calibration parameters and the captured data in the $X - Y$ directions, the upper surfaces of two wooden blocks with sizes of $125 \times 84 \times 36$ mm (block a) and $74 \times 37 \times 18$ mm (block b) were measured with the incidence angle at 0 in our experiments. Another wood block with a size of $116 \times 59 \times 29$ mm (block c) was measured with the incidence angle at 30^0 to check the accuracy in the Z direction and the effect of the incidence angle. The captured range images and corresponding calibrated 3D data transformed by the calibration parameters are similar to the images shown in Fig. A.12.

Based on the calibrated 3D data, the sizes of the block can be estimated as follows. For each of the tested blocks, first, four corner points on the same plane are detected heuristically from the corresponding VRML file of calibrated 3D data. The distances between these points can be calculated from the VRML file because the calibrated 3D data consists of Euclidean coordinates of all the data points. These distances, actually, represent the exact sizes of the block because the Euclidean coordinates in the calibrated data have a physical unit, mm in this case, during



(a) (b)

Figure A.14: Captured range and corresponding calibrated 3D data. (a) Range image of the upper flat surface, the surface is parallel to the ground and (b) calibrated 3D data of the surface.

the calibration process. In this way, the size of the block a is estimated as 125.4295×84.3160 mm. The size of the block b is estimated as 74.8173×36.9495 mm. The size of the block c in the z axis is estimated as 29.212 mm. Compared with the ground truth values of the sizes of the three blocks, the accuracy of this camera and the calibration is approximately 0.5 mm and the measurement error is approximately 0.7%. In addition, the accuracy of the captured data is insensitive to an incidence angle less than 30^0.

A.5 Summary

Based on the characterization results, this section summarizes how the performance of the smart range sensor can be optimized including how the sensor should be used appropriately and how the accuracy of the measured data can be improved in various circumstances. The principles used in setting up the ranging system and selecting the parameters in the horizontal threshold algorithm are summarized as follows.

- The major parameters of the algorithm are chosen heuristically depending on the size of the object measured, the accuracy required, and the speed of the conveyor belt. Therefore, a training process is needed to choose the parameters.
- The camera needs to be calibrated when the system setup or the speed of the conveyor belt is changed. However, to capture more accurate images, a new camera calibration is recommended for each new acquisition.
- The size of measured objects should be comparable to the distance between the baseline and the objects to obtain correct range measurements. In addition, the length of the baseline should also be comparable to the distance between the baseline and the object.

Table A.1: Factors affecting range measurements and corresponding corrections.

Factor	Sensitivity	Suggested Correction
Temperature	low	warm up 60 mins
Light	high	achieve the least lumen of the light
Reflectance	low	use a bandpass filter
Color	high	avoid black surfaces
Incidence Angle	low	none

The sensitivity of factors affecting range measurements are summarized in Table A.1. This table also indicates that noise present in the captured range data is mainly caused by environmental light and the surface color of scanned objects. Therefore, in order to obtain more accurate data and optimize the performance of the sensor, these factors should be reduced as much as possible. The accuracy of the camera is approximately 0.5 mm, and the error present in the 3D data is approximately 0.7%. Based on the characterization results and accuracy evaluation, this range sensor is believed suitable for indoor, small object-related tasks such as object modeling, representation, and recognition. Although the camera investigated in this work is designed mainly for indoor use it is possible to take outdoor images with an optical filter added to the lens. The filter should be chosen according to the bandwidth of the laser used. Characterization of this sensor in various outdoor environments is an ongoing task.

In this chapter, a general concept on how the performance of a laser range sensor can be evaluated and maximized by experimentally characterizing a smart range camera is presented. By characterizing a sensor comprehensively, the optimal configuration including setup and appropriate applications can be derived. In addition, the factors that affect the accuracy of the captured data and their quantitative measurements can also be obtained during the characterization process. Therefore, the performance of a sensor can be maximized correspondingly based on its characterization results, e.g. the best configuration should be obtained and negative factors should be avoided or reduced. In summary, we demonstrate that experimental characterization of a sensor is indispensable for evaluating, and ultimately, optimizing the sensor performance, and how this can be achieved.

Vita

Yan Zhang was born on May 17, 1973 in China. After completing her higher secondary education in the city of Zhengzhou in July 1990, she entered the Department of Automatic Control Engineering, Huazhong University of Science and Technology, where she was awarded a Bachelor of Science degree in June 1994. Fascinated by the area of image processing and pattern recognition, she began her graduate study in the Institute of Pattern Recognition and Artificial Intelligence (PRAI) at Huazhong University of Science and Technology, and was awarded a Master of Science degree with honors in June 1997. She worked as a graduate researcher in PRAI for one year after graduation.

In August 1998, she enrolled in the Ph.D. program in the Department of Electrical Engineering at the University of Tennessee, Knoxville. She joined the Imaging, Robotics and Intelligent Systems laboratory as a graduate research assistant and has been working under the supervision of Dr. Mongi A. Abidi. She will complete her graduate study and achieve the Ph.D. degree in August 2003 with a specialization in image processing and computer vision. While not thinking about image processing or other academic matters, she likes to enjoy music and reading.